CS Exit Ethiopia

Revision Outlines

Computer Science Exit Examination

1 selected course | 18 November 2026

Selected courses

01. Network and System Administration

https://csexitethiopia.com

abelasnake2003@gmail.com | Telegram: @Abechu7

Course contents

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

Network and System Administration 3

Network and System Administration

Revision Outlines

Roles, principles and change discipline | Exam focus

- Administrative roles and qualities
  - Systems: hosts/accounts/storage/software/availability
  - Networks: connectivity/addressing/routes/devices/performance
  - Security: controls and policy
  - Overlapping account/patch/training duties
  - Job titles ≠ exclusive duties
  - Patience, communication, ethics
  - Technical understanding and organization
  - Beyond command knowledge
- Administration and measured change
  - Documentation, standardization, least privilege, automation
  - Record current state and intended change
  - Rollback and validation
  - Maintenance: service impact and urgency
  - Calendar date alone insufficient
  - Fault management: restore failures
  - Performance: measure service behavior
  - Configuration: track settings
  - Security: control access
  - Accounting: track usage

Network foundations for administrators

- Client/server and peer roles
  - Client: requests; server: provides
  - Peer-to-peer: both roles
  - Server may request DNS/backups
  - Dedicated server ≠ never client
  - Network OS ≠ dedicated-server-only

- Clients require networking support
- Centralized service software versus participant OS

• Forwarding, transport and diagrams
- Switching: LAN frames by MAC
- Routing: inter-network IP packets
- TCP: ordered reliable byte stream
- UDP: datagrams, no equivalent guarantee
- Ports: application endpoints, not connectors
- Diagram switches, links, subnets, gateways
- Service hosts and dependencies
- Topology shape ≠ security guarantee

Addresses, prefixes, gateways and loopback

• IPv4 addressing rules
- IPv4: 32 bits
- CIDR prefix determines network
- Network=address AND mask
- Remote subnet: route/gateway
- Private: 10/8,172.16/12,192.168/16
- Private ≠ directly Internet-routable
- IP/class label alone ≠ subnet

• Loopback meaning and test limits
- 127.0.0.1: conventional local IPv4 loopback
- 127/8 reserved
- Tests local protocol processing
- No cable/switch/remote reachability proof
- Local service use
- Multiple loopback descriptions may fit
- Multiple 127/8 choices may fit

• Worked example Subnet-boundary example
- 192.168.2.70/26
- Block size: 64 addresses
- Network .64; broadcast .127

- Ordinary hosts .65-.126
- Gateway requires appropriate reachable route
- NAT ≠ missing-route repair

DHCP exchange and lease lifecycle | Exam focus

- DHCP initial allocation
  - Addresses and configuration options
  - Discover→Offer→Request→ACK
  - Scope/pool and allocated leases
  - Exclusions and reservations
  - Mask, gateway, DNS options
  - Dynamic lease: specified interval
  - Lease ≠ forever
- Lease renewal and reuse
  - Renew before expiry
  - Failed renewal: later rebinding
  - Disconnected host ≠ perpetual renewal
  - Expiry: address eligible for reuse
  - Server-policy conditions
  - Reconnection: validate/renew/new allocation
  - Expiry alone ≠ persistent failure
  - Reservation: known client, predictable address
  - DHCP configuration retained
- Worked example Expired disconnected-client example
  - Expiry while other clients allocate
  - Old address may be reassigned
  - Validate configuration before reuse
  - No unvalidated old-address resumption

DHCP deployment and diagnosis | Exam focus

- DHCP deployment rules
  - Subnet-matching scope
  - Exclude manual server/router addresses

- Correct configuration options
- Relay: broadcasts across router
- Static and DHCP coexist
- Avoid address conflicts
- Less repetitive administration
- Availability/security: deployment-dependent
- Rogue-reply protection
- Location change ≠ guaranteed connectivity

• DHCP returning-client diagnosis
- Inspect current lease/address
- Server/relay reachability
- Free pool capacity and logs
- New devices: possible pool exhaustion
- Exhaustion requires evidence
- Link-local fallback: 169.254.2.33 example
- Same-link communication
- Not ordinary routed Internet access
- 0.0.0.0: unspecified
- 255.255.255.255: limited broadcast

DNS resolution and diagnostic evidence | Exam focus

• DNS record roles
- Forward: name→address
- A: IPv4; AAAA: IPv6
- Reverse: address→name
- PTR: reverse namespace
- CNAME: alias
- MX: mail exchangers
- NS: authoritative name servers
- Forward/reverse: independent existence
- Not necessarily perfect inverses

• DNS caching and diagnosis
- Resolver caching: TTL

- nslookup name
- Explicit DNS-server query
- Configured resolver versus authoritative answer
- Address ping succeeds, lookup fails
- Naming/configuration trouble suggested
- Ping by name: resolved address possible
- nslookup: direct DNS diagnostic
- Resolving includes forward resolution

Commands and observable state | Exam focus

- Address-configuration commands
  - Windows ipconfig: address/mask/gateway
  - ipconfig /all: complete adapters
  - Physical/MAC, DHCP, DNS information
  - /renew: request renewal
  - /release: dynamic address release
  - /flushdns: client DNS cache clearing
  - Linux ip addr: addresses
  - Linux ip route: routes
  - Persistent configuration: distribution-specific
- Connection and path diagnostics
  - netstat: endpoints/protocols/TCP states
  - netstat -ano: numeric addresses/ports/PIDs
  - Literal netstart ≠ netstat
  - Ping: IP reachability probes
  - traceroute/tracert: responding hops
  - TTL/hop-limit expiry
  - ICMP filtering: replies may fail
  - Missing reply ≠ service-down proof
  - nslookup: DNS inspection

Repeatable troubleshooting procedure | Exam focus

- Evidence-led troubleshooting sequence

- Symptom, scope, time, recent changes
- Configuration, baseline, logs
- Testable hypothesis
- Least disruptive explanation first
- Isolate failing component
- Targeted fix; verify user service
- Document results
- One relevant factor at a time
- Reboot may hide evidence
- Reboot ≠ diagnosis

• Troubleshooting approaches
- Bottom-up: physical link first
- Top-down: application errors first
- Divide-and-conquer: test middle layer
- Narrow working/failing boundary
- Divide-and-conquer ≠ random trial-and-error
- Reboot/update/reset/scan: context-dependent actions
- No fault evidence: no unique ranking

• Worked example Scope-based diagnosis examples
- One client fails: inspect client
- Link, address and policy first
- All name lookups fail
- Server IPs still reachable
- Inspect shared DNS/configuration first
- Cable replacement: unsupported by naming-only failure

Connectivity diagnosis by layers | Exam focus

• Layered connectivity checklist
- Power, cable/connector, link state
- NIC and driver state
- VLAN membership
- Address/prefix and local gateway
- Route, DNS, application port/authentication

- Recognized NIC: driver absence less likely
- Recognition ≠ working physical link
- Ethernet: commonly RJ45
- RJ11 telephone jack ≠ expected Ethernet

• Interpreting connectivity symptoms
- Local works, remote fails
- Inspect gateway/routes/filtering
- Application works, ping fails
- ICMP filtering plausible
- Port refused: destination reachable enough
- Inspect service/listener policy
- Timeout: loss/filtering/down host possible
- Timeout requires further evidence

• Worked example Post-replacement NIC example
- Recognized/enabled adapter; no link light
- Try known-good Ethernet cable
- Try known-good switch port
- Recognition ≠ one defective-cable diagnosis

Service catalog and remote access | Exam focus

• Service protocol roles
- DNS: names; DHCP: host configuration
- SMTP: mail send/relay
- IMAP/POP: mail retrieval
- FTP: named file-transfer protocol
- HTTP also transfers files
- File transfer ≠ exclusive FTP description
- HTTP/HTTPS: web resources
- SSH: encrypted remote login
- SFTP: file transfer over SSH
- SFTP ≠ FTP/FTPS

• VPN, media and service administration
- VPN: private connectivity over another network

- Secure tunnel: encryption/authentication
- Protocol/configuration conditions
- RTP: real-time timing/sequence information
- Often over UDP
- RTP ≠ timely-delivery guarantee
- Network printers: shared services
- Service accounts, permissions, configuration
- Monitoring and recovery

Routing, NAT and port translation

- Routing versus NAT/PAT
  - Routing: destination-prefix next-hop selection
  - NAT: address rewriting
  - PAT/NAPT: many-to-one address-and-port translation
  - Distinct bindings on one public address
  - Two private sockets→ports 40001,40002
  - Replies translated to correct clients
- NAT scope and limits
  - Private IPv4→public networks: common use
  - NAT ≠ gateway/route/firewall/authentication replacement
  - One-to-one NAT
  - Destination translation
  - NAT ≠ always many-to-one
  - Public LAN addresses: NAT not automatic
  - Unsolicited inbound service: explicit mapping possible
  - Filtering requirements

Proxies, gateways and QoS

- Proxy and gateway functions
  - Proxy: client/backend representative
  - Cache, filter, mediate application requests
  - Default IP gateway: next-hop router
  - Application gateway: possible protocol translation

- Gateway ≠ mandatory modem/router combination
- Functions may coexist on one machine

• QoS and mixed-category questions

- QoS: classify, queue, prioritize traffic
- Low-delay/limited-loss objectives
- Operates across layers
- Supports applications
- QoS ≠ exclusively application-layer service
- New-service installation: administrative action
- Email configuration: administrative action
- Proxies/agents: components
- Mixed taxonomy: unique exclusion needs scope

Workgroups, domains and directories | Exam focus

• Workgroup versus domain

- Workgroup: per-machine local accounts
- Local credentials and per-host sharing
- Domain: centralized identities/policies
- Directory infrastructure
- Administrative arrangement ≠ physical topology
- Recommended size ≠ hard protocol limit
- HomeGroup: historical Windows sharing
- HomeGroup ≠ AD domain

• Domain controllers and availability

- Controller: AD DS
- Domain authentication and directory replication
- Redundancy: dependencies/failure domains matter
- Two virtual controllers, one physical host
- Shared host failure ≠ independent availability
- Provisioning: addressing, DNS, time
- Supported OS/hardware
- Version minima ≠ universal architecture rules

LDAP and secure directory access

- LDAP operations and transport protection
  - Entries/attributes: distinguished names
  - Bind: session authentication
  - Search: matching entries
  - Compare: attribute-value test
  - Add/delete/modify: entry changes
  - Modify-DN: rename/move
  - Unbind: end session
  - StartTLS: upgrade connection where supported
  - LDAPS: TLS from setup
  - Protect credentials; verify server identity
- Directory ports and security caveats
  - LDAP: commonly TCP 389
  - LDAPS: commonly TCP 636
  - Acronym/port ≠ secure authentication
  - AD also uses DNS/Kerberos
  - LDAP ≠ entire directory product
  - LDAPS-retired-with-LDAPv2 claim: unreliable
  - Modern TLS-protected LDAP distinction

AD structure and delegation

- Directory objects and logical hierarchy
  - Objects: users/computers/groups/resources
  - Schema-defined attributes
  - Security principals: SIDs
  - Display name ≠ sole security identity
  - Domain: objects plus replicated database
  - Tree: contiguous DNS namespace domains
  - Forest: grouped trees
  - Shared schema/configuration
  - Global-catalog framework
- OUs, groups and schema restrictions

- OUs: object organization
- Delegation and Group Policy scopes
- OUs ≠ separate forests/security boundaries
- Groups: access assignment
- OUs: policy/delegation structure
- Schema changes: broad propagation, care
- Ordinary objects: deletion possible
- Schema-definition restrictions: special scope
- Schema restrictions ≠ all object restrictions

Sites, replication and trusts

• AD sites and replication
- Sites: grouped IP subnets
- Network locality model
- Controller selection
- Replication traffic management
- Sites ≠ domains/OUs
- Replicate changes among controllers
- Intrasite notification-triggered pulls
- Intersite schedules/link costs
- Replication ≠ prior-good-state backup

• Trust direction and permissions
- A trusts B
- B identities authenticate for A resources
- Permissions still required
- Trust ≠ automatic file access
- One/two-way direction
- Transitive/nontransitive extension
- Direction and transitivity: independent
- Explicit trust ≠ fixed direction/transitivity
- Trust type/configuration determines behavior
- State direction before drawing arrows

Accounts, rights and password diagnosis | Exam focus

- Account lifecycle and access models
  - Named accounts
  - Only needed groups/rights
  - Disable departed users
  - Audit privileged activity
  - Authentication: verify account
  - Authorization: permitted operations
  - User right: remote logon
  - File permission: object-specific
  - Groups easier to review than individual grants
- Single-user password-change diagnosis
  - Inspect cannot-change-password account setting
  - Relevant policy/permission
  - Minimum age, password rules, account state
  - Plausible cause ≠ proven conclusion
  - One failure ≠ infrastructure-wide diagnosis
  - Administrators membership ≠ inherent change prohibition
  - Privileged reset ≠ user-initiated change

Windows permissions and effective access | Exam focus

- Windows permission evaluation
  - Read, write, read/execute
  - Modify and full control
  - Granular rights: delete
  - Inherited versus explicit entries
  - Group membership and deny entries
  - Object scope
  - Effective-access view: real account
  - Informal permission-label addition insufficient
  - Network access: share plus filesystem constraints
- Rights and access distinctions
  - Delete: valid Windows permission

- OS-unspecified Delete exclusion: false
- Modify ≠ full control
- Full control: permission/ownership capabilities
- Data access ≠ logon rights
- Read ≠ write
- Read ≠ parent-folder administration

Unix rwx and octal permissions | Exam focus

- Unix permission interpretation
  - Nine bits: owner/group/other triplets
  - Read=4; write=2; execute=1
  - Sum enabled bits per octal digit
  - Leading ls -l character: file type
  - Directory read: list names
  - Directory execute: traverse/search
  - Directory write: change entries
  - Other checks apply
  - Deletion: usually parent-directory permissions
  - Sticky-bit/ACL rules possible
- Worked example Octal and damaged-string examples
  - -rwxr--r--: ordinary file
  - Owner7/group4/other4 → 744
  - 755 → rwxr-xr-x
  - Damaged bank string: -rwxr-r-
  - Missing hyphens: no guessing
  - Confirm all nine bits before choice

Linux paths, filesystems and checks

- Linux directory roles
  - /: filesystem root
  - /root: administrator home
  - /etc: configuration
  - /home: ordinary user homes

- /var: variable data/logs
- /dev: device nodes
- /proc: kernel/process information
- /boot: boot material
- /tmp: temporary files
- Cleanup/mount layouts vary
- lost+found ≠ every partition guarantee
- Reboot ≠ every temporary directory erased

• Filesystems, checking and mounting
- ext2: Linux filesystem
- ext4: later filesystem
- HPFS: historical OS/2 association
- Swap: backing space, not ordinary filesystem
- fsck: filesystem-specific checks/optional repairs
- fsck ≠ init/halt/inittab
- Supported offline/safe repair procedure
- Mount: filesystem attached to directory
- Storage administration: partitioning/mounts/quotas/capacity

• Filesystem origin versus driver support
- Foreign filesystem access possible
- HPFS access: available driver required
- Origin ≠ Linux support
- Swap: paging area
- Native Linux filesystem: ext2
- Native-Linux scope must be explicit
- Broad Linux-use wording: ext2/HPFS ambiguity

Profiles, registry and persistent configuration

• User profile variants
- Local: settings on one client
- Roaming: server synchronization at logon/logoff
- Supported user settings/data
- Large profiles: slower logon/logoff

- Folder redirection: avoid repeated folder copying
- Central profile backup: policy-dependent
- Cached profile ≠ independent local security account

• Windows registry

- Hierarchical configuration database
- Keys, subkeys, typed values
- HKLM: machine settings
- HKCU: current-user view
- Registry ACLs
- Historical root handles: version-dependent
- Record/export relevant configuration before change
- Authorized change; verify service behavior

Performance monitoring and bottlenecks

• Performance diagnosis before hardware changes

- Baseline and interval sampling
- Workload comparison and symptom correlation
- CPU utilization/run queues: processing pressure
- Low free memory: weak evidence alone
- Reclaimable cache
- Persistent paging plus latency: memory pressure
- Storage capacity ≠ I/O latency
- Nearly empty disk: saturation possible
- Nearly full disk: light load possible

• Network measurements and collection methods

- Interface bytes/second
- Errors, drops, delay/loss
- Utilization versus link rate
- Capacity: maximum available rate
- Throughput: observed rate
- Logs: security and performance evidence
- Counters: measured events
- System calls/APIs/log parsers: collection routes

- Methods need not be exclusive

- Worked example Performance calculation and diagnosis examples
  - 80 MB/10 s=8 MB/s
  - Approximately 64 Mb/s before overhead
  - CPU 95% plus growing runnable queue
  - Suggested CPU pressure
  - Disk 95%-full: capacity action
  - Low latency ≠ sufficient free capacity

Monitoring, capture and assessment tools

- Monitoring-tool capabilities
  - Task Manager: processes/resource use
  - Resource Monitor/Performance Monitor: detailed counters
  - Wireshark: packet capture/dissection
  - Nmap: host/port/service discovery
  - Nessus: vulnerability/configuration assessment
  - Snort: detection and alerts
  - Snort inline: prevention possible
  - Snort: sniffing/logging roles
  - Packet capture: not exclusive
  - Wireshark: conventional analyzer choice
- Detection, prevention and evidence limits
  - IDS: detect/alert
  - IPS: inline blocking
  - Scan: potential weaknesses
  - Scan ≠ proven exploitation
  - Packet/open port/alert: context required
  - Usage monitoring ≠ access control
  - Usage monitoring ≠ encryption

Backups, restore and continuity | Exam focus

- Backup purpose and independence
  - Recover previous data

- Deletion, compromise, corruption
- Failed upgrades/migrations; hardware loss
- Authorization: current access, different purpose
- Mirror/replica copies deletions/corruption
- Replica ≠ versioned independent backup
- Protected separate failure domains
- Retention and restore tests

• Backup types and recovery objectives
- Full: chosen dataset
- Incremental: since last backup
- Differential: since last full
- Incremental restore: full plus subsequent incrementals
- Differential restore: full plus latest differential
- RPO: tolerable lost-data time
- RTO: tolerable restoration time
- Compression: reduced storage
- Scheduling: operation timing
- Restore testing: recovery usability

RAID layouts, capacity and selection

• RAID layout mechanisms
- RAID0: striping, capacity/throughput
- RAID0: no disk-failure tolerance
- RAID1: mirroring
- Two-disk mirror: one disk's usable capacity
- Two-disk mirror: survives one disk failure
- RAID2: bit striping/Hamming redundancy
- RAID3: byte striping/dedicated parity
- RAID4: block striping/dedicated parity
- RAID5: block striping/distributed parity
- RAID6: distributed dual parity
- RAID6: two disk failures tolerated
- RAID10: striped mirrored pairs

- RAID10 survival: failed-disk placement dependent

• RAID capacity and selection rules
  - N equal disks, each capacity D
  - RAID0=ND
  - Paired RAID1/10≈ND/2
  - RAID5=(N−1)D
  - RAID6=(N−2)D
  - Hardware controller resources/cache: possible offload
  - Hardware ≠ universally faster software RAID
  - Rebuild bandwidth and parity-write overhead
  - University/bank size ≠ level selection
  - Striping/parity ≠ uniquely RAID5
  - Distributed block parity: RAID5 distinction

• Worked example Four-disk capacity example
  - Four 2 TB disks
  - RAID0=8 TB; RAID10=4 TB
  - RAID5=6 TB; RAID6=4 TB
  - Metadata ignored
  - Failure tolerance, workload, capacity, recovery
  - RAID layouts ≠ backups

Security policy development and purpose | Exam focus

• Security-policy lifecycle
  - Required behavior, scope, accountability, enforcement
  - Assets and service objectives
  - Assess threats and weaknesses
  - Choose controls
  - Exceptions and incident procedures
  - Approve and communicate
  - Monitor and periodic review
  - CIA: confidentiality, integrity, availability
  - No single universal primary objective

• Policy requirements and implementation

- Password controls and approved encryption
- Restricted administrative access
- Backups and patch deadlines
- Firewall rules: partial implementation
- Unrestricted wireless conflicts with internal control
- Isolated intentional guest service: exception
- Passive system use ≠ policy purpose
- Coordinated action and reduced risk
- Policy ≠ all attacks fail

Hardening, controls and vulnerabilities | Exam focus

• Hardening and risk
  - Remove unnecessary services
  - Supported patches
  - Restricted accounts/permissions
  - Appropriate port filtering
  - Unused known-port blocking: possible hardening
  - Blocking useful ports: availability destroyed
  - Weak passwords/unpatched exploitable software: vulnerabilities
  - Attacks exploit weaknesses
  - Risk: likelihood with impact
• Control functions and patch discipline
  - Preventive: restrict harmful actions
  - Detective: expose harmful actions
  - Recovery: restore service
  - Accounts/backups/configuration/traffic analysis: possible controls
  - Stated objective determines control relevance
  - Vulnerability scan: weakness patterns, not bandwidth
  - Test/stage patches
  - Prepare rollback; deploy; verify
  - Testing/installation: both administrator duties

Encryption, hashes and script secrets

- Cryptographic purpose distinctions
  - Encryption: confidentiality
  - Correct key/nonce use
  - Authenticated encryption: integrity/authenticity too
  - Unspecified encryption ≠ both guarantees
  - Password hash: one-way verification
  - Hash ≠ reversible credential store
  - Recoverable credential encryption: protected separate key
- Script-secret handling
  - No reusable secrets in source/logs
  - Protected credential service
  - Restricted runtime source
  - Environment variables/configuration files: storage/transport
  - Neither automatically encrypted
  - Service authentication: actual credential/token often required
  - One-way hash ≠ usable replacement
  - Verification versus recoverable service credentials

Incidents, evidence and reports | Exam focus

- Incident-response lifecycle and reporting
  - Prepare; detect/analyze
  - Contain; remove cause
  - Restore service; learn
  - Investigation: reactive response
  - Patching/backups: normally preventive/preparatory
  - Timestamps and affected systems
  - Evidence, impact, actions
  - Responsible people and recovery results
  - Report may include policy violations
  - Incident scope broader than violations
- Attacks and evidence handling
  - DoS: availability harm

- DDoS: distributed sources
- Phishing: user deception
- Brute force: credential attempts
- Interception: communications read/altered
- Correlate logs by time/identity/context
- Preserve evidence before resets
- Suspicious event ≠ confirmed compromise
- Record facts and uncertainty

Segmentation, NAC and remote administration | Exam focus

- Segmentation and admission control
  - Separate system groups
  - Routing/filtering: allowed flows
  - VLAN membership ≠ complete security
  - NAC: device/user admission assessment
  - Reject/quarantine unauthorized/noncompliant devices
  - Authentication: prove identity
  - Access rules: allowed resources
- Protected remote administration
  - SSH/secured management services
  - Authentication and protection
  - Restricted source access; auditing
  - VPN: defined protected-connectivity model
  - VPN ≠ packet analyzer
  - Firewall: policy-based permit/block
  - Firewall filtering ≠ inherent encryption
  - Management route/firewall changes: connectivity-loss risk
  - Plan for remote lockout

Scripting inputs and reusable operations | Exam focus

- Administrative script building blocks
  - Repeatable automation
  - Scripts ≠ inherently faster/secure

- Python, shells, PowerShell
- HTML: markup
- Define inputs and outputs
- Variables: state
- if/elif/else: decisions
- Loops: repetition
- Functions: reuse
- Python indentation: blocks
- ==: comparison; =: assignment

• Predictable and safe reruns
- Compute desired state
- Compare current state
- Change only differences
- Small dry run before mutation
- Read-only reporting: avoid unnecessary privilege
- Report failures
- Meaningful exit status
- No unconditional success output

• Worked example Threshold-function trace
- def warn_percent(values, limit)
- Loop unpacking: name,percent
- Condition: percent>=limit
- Inputs: root=92,data=40,limit=90
- Only root meets threshold
- print: space-separated output
- root high 92

Python subprocess command launching | Exam focus

• subprocess module and run contract
- Python module ≠ universal shell command
- run: child launch and result
- capture_output: stdout/stderr collection
- text=True: text decoding

- timeout: bounded wait
- check=True: nonzero exit raises
- Argument list: avoid shell interpretation
- List arguments versus shell-string quoting

• Command versus language abstraction

- Echo: print
- grep: text search
- ipconfig/ifconfig: network reporting/configuration
- Neither general Python child-process abstraction
- Generic command question: language required

• Worked example subprocess example and failures

- subprocess.run: ip addr show
- text=True; capture_output=True
- check=True; timeout=5
- print(result.stdout)
- Linux; iproute2 installed
- Missing program: OSError
- Nonzero status: CalledProcessError
- Timeout: TimeoutExpired

File I/O, parsing and updates | Exam focus

• Configuration-processing tools

- File I/O: obtain/persist configuration
- String operations: parse values
- Regex: recognize patterns
- Broad useful-concept question: multiple answers
- JSON/INI: format-specific parser
- Avoid blind global substring replacement
- Explicit text encoding
- with: closure even after failure

• Configuration-update sequence

- Validate desired output
- Preserve known-good copy

- Temporary file: same filesystem
- Atomic target replacement where supported
- Check permissions and syntax
- Reload affected service
- Verify; retain rollback
- Repeated append ≠ idempotent

• Worked example JSON validation example
- with open: settings.json, encoding=utf-8
- json.load(handle)
- port=config["port"]
- isinstance(port,int) required
- 1<=port<=65535
- Invalid type/range: ValueError
- Output: validated port and value
- Reporting only; no system-file change

Arguments and targeted error handling | Exam focus

• Command-line parsing
- Named typed values and usage
- Python argparse
- Positional arguments: required inputs
- Flags: options
- Validate ranges and paths
- Unchecked input ≠ executable shell text

• Worked example Argument/error-handling trace
- ArgumentParser and parse_args
- logfile: Path
- --limit: int, default 10
- limit<1 → p.error
- read_text: UTF-8
- [:limit]: characters, not lines
- Catch OSError/UnicodeError
- Exception skips remaining try

- p.exit(1): read-failed message
- Specific catches expose unexpected programming errors

Regex analysis and log rotation | Exam focus

- Regex contracts and log rotation
  - Text patterns, not encryption/compression/rotation
  - search: match anywhere
  - fullmatch: entire string
  - Raw strings: backslash-heavy patterns
  - Capture needed fields only
  - Explicit unmatched/malformed-line handling
  - Rotation: bounded size, historical segments
  - Parsing and rotation: complementary
- Log-analysis evidence
  - Count requested event
  - Correlate timestamps and hosts
  - No matches ≠ unreadable log
  - Security and performance uses
  - Neither purpose universally exclusive
- Worked example Regex status-count example
  - re.compile: status=(\d{3})\b
  - Three digits and word boundary
  - pattern.search per line
  - if match: group(1)
  - counts.get(code,0)+1
  - 200 and 503: one each
  - Noise deliberately ignored

CPU and memory measurements | Exam focus

- Interval CPU measurement
  - Cumulative counters: two samples required
  - Lifetime total ≠ current utilization
  - Linux /proc/stat: CPU categories

- Define idle categories
- Avoid guest-field double counting
- Busy_fraction=1−Δidle/Δtotal
- Chosen nonbusy: idle+iowait
- Iowait interpretation limitations

• Available-memory measurement
- /proc/meminfo: MemAvailable
- Estimate without swapping
- MemFree excludes reclaimable memory
- Windows Performance Counters/OS APIs
- Analogous metrics, different interfaces
- System calls/counters/log parsing: complementary
- Only-technique claim requires platform/API

• Worked example Measurement calculation example
- Δtotal=1000 ticks; Δidle=250
- Busy≈75%
- MemTotal=8192 MiB; MemAvailable=2048 MiB
- Approximate unavailable/used=75%
- MemAvailable ≠ exact-allocation success guarantee

Backup scheduling and restore verification | Exam focus

• Recurring-job scheduling
- Unix: cron/systemd timers
- Windows: Task Scheduler
- User crontab: minute,hour,day-of-month,month,day-of-week,command
- Absolute paths and known environment
- Protected inputs and logs
- Bounded runtime
- Lock unsafe overlapping jobs
- Interactive PATH/CWD/credentials: not assumed

• Backup verification and tool purposes
- Compression: data packaging
- Scheduling: operation timing

- Check exit status and archive contents
- Restore sample into isolated destination
- Command success ≠ usable backup proof
- Script version control: review aid
- Version control ≠ data backup

• Worked example Cron entry interpretation

- 0 2 * * *
- Daily 02:00, scheduler timezone
- Runs as crontab owner
- /usr/bin/python3 /opt/backup.py /srv /bak
- Source/destination arguments
- >>/var/log/bak.log: append output
- 2>&1: stderr joins stdout
- Command kept on one line
- Paths/permissions/logging/program: preconfigured
- User crontab: no username field
- System crontab: username possible

Constructing a bounded backup script | Exam focus

• Backup-script structure and limitations

- Separate parsing, preconditions, archive creation, reporting
- Source directory required
- Destination outside source
- Unique output names
- Failed partial ≠ completed backup
- Application consistency and capacity checks
- Retention and independent protected copies
- Teaching script: directory packaging
- No silent scheduled-job installation

• Worked example Backup-script trace and verification

- argparse source/destination: Path
- resolve(): absolute source/destination
- is_dir(): existing directories required

- dst==src or src in dst.parents
- Reject destination inside/equal source
- UTC timestamp name; microseconds
- shutil.make_archive: gztar, root_dir=src
- OSError: remove partial .tar.gz
- unlink(missing_ok=True)
- p.exit(1): failure report
- print completed archive path
- Verify archive members
- Restore into different empty directory
- Compare selected files