CompTIA Autocode+

Certification Exam
Pre-draft Exam Objectives
Exam Number: AT0-001

- Pre-draft Exam Objectives summarize the tasks and skills identified in the Job Task Analysis (JTA) workshop that provide directional information about the upcoming exam version.
- The Draft Exam Objectives will replace the Pre-draft Exam Objectives after approximately two months when the skills have been peer-evaluated and validated through a JTA survey of job role practitioners.
- Pre-draft Exam Objectives may contain typos and errata that will be corrected during the development process.
- CompTIA will not accept feedback on the Pre-draft Exam Objectives document. If errors are found, please wait until the Draft Exam Objectives are posted, and then provide feedback using the Draft Exam Objectives Feedback form.

1.0 Automation Coding Concepts

1.1 Given a scenario, use code to support automation.

- Variables
  - Scope
    - Global
    - Local
- Conditional statements
- Iterations
- Parameters
- Functions
- Application logs
- Regular expressions (RegEx)
- Data types
  - Primitive
  - JavaScript Object Notation (JSON)
  - YAML Ain't Markup Language (YAML)
- Command-line tools
  - `grep`
  - `awk`
  - `sed`
  - `jq`
  - Cmdlets
- Dependency management
  - Dockerfile
  - `requirements.txt`
  - Packages
  - Libraries

1.2 Given a scenario, use source control techniques to version code.

- Git commands
  - Remote
    - `fetch`
    - `pull`
    - `push`
  - Local
    - `add`
    - `commit`
  - Config
- Semantic versioning
  - Pre-release versions
  - Major vs. minor releases
  - Filtering techniques
- Branching strategies
  - Release branching
  - Feature branching
- Branch naming conventions
- Commit life cycle
  - Hooks
  - Formatting
  - Linting
  - Detecting sensitive data
  - Comments

1.3 Explain concepts related to infrastructure as code (IaC).

- Reusability

- Immutability
- Declarative vs. imperative
- Idempotency

1.4 Given a scenario, troubleshoot common issues with the code life cycle.

- Syntax errors
  - Undefined variable errors
- Runtime errors
- Git errors
  - Merge conflicts
  - Authentication issues
  - Detached HEAD

2.0 System Configuration

2.1 Given a scenario, use configuration management techniques to deploy a solution.

- Drift detection and remediation
- State management
- Workload configuration
  - Certificates
  - Access control lists (ACLs)
- Virtual network infrastructure configuration
  - Subnets
  - Routers
  - Load balancers
  - Firewalls
  - Route tables
  - Domain Name System (DNS)

2.2 Compare and contrast basic approaches to automation.

- Remote vs. local
  - Remote
    - Windows Remote Management (WinRM)
    - Secure Shell (SSH)
    - Agent-based
  - Local
    - Command-line script
    - Background/foreground
- Declarative vs. imperative
  - Migrations
  - Ephemeral environments
  - Persistent environments
  - Script writing
- Provisioning techniques
  - Push
  - Pull
- Cloud-based events
  - Notification-based events
  - Messaging queues
  - Function as a Service (FaaS)

2.3 Given a scenario, interact with Representational State Transfer–compliant (RESTful) systems to perform create, read, update, and delete (CRUD) operations.

- Headers
- Methods
  - `GET`
  - `POST`

    - `DELETE`
    - `PATCH`
    - `PUT`
- REST application programming interface (API) tools
    - `curl`
    - `wget`
    - `postman`
- Response codes
    - 2XX
    - 3XX
    - 4XX
    - 5XX

2.4 Given a scenario, troubleshoot common issues with system configuration.

- API communication failures
    - Agent communication issues
- Application failures
- Certificate issues
- Syntax errors specific to configuration files
    - INI
    - YAML
    - JSON

3.0 Continuous Integration

3.1 Explain environmental factors related to continuous integration (CI) management.

- Secrets management
    - Encrypted key vault
    - Revocation
    - Lease management and expiration
    - Tokenization
    - Zero Trust
    - Dynamic secret rotation
- Artifact management
    - Artifact registry
    - Authorization
    - Authentication
    - Versioning
    - Immutability
    - External artifact repository
    - Software bill of materials (SBOM)
- Task runners
    - Environment provisioning

3.2 Explain workflow management concepts in CI.

- Orchestration techniques
    - Parallel vs. sequential
    - Step dependencies
    - Conditional execution
    - Automated rollbacks
    - Failure handling
- Pipeline-enhancing concepts
    - Security scanning
    - Containerization
    - Code quality scanning
    - Artificial intelligence (AI)–based log analysis

3.3 Given a scenario, analyze configurations to manage basic automation pipelines.

- Hooks and triggers
  - Webhooks
  - Chat-based operations (ChatOps)
- Pipeline triggers
  - Push trigger
  - Manual trigger
  - Scheduled trigger
  - Git variables
    - Tag
    - Branch
    - Commit
- Pipeline definition
  - Jenkinsfile
  - `gitlab-ci.yml`
  - `github-actions.yml`

4.0 Continuous Delivery (CD)

4.1 Given a scenario, implement techniques of continuous delivery (CD).

- Delivery methods
  - Canary
  - Blue-green
  - Rolling
  - In-place
- Quality assurance (QA) testing
  - Load testing
  - Regression
  - Integration
- Validation and remediation
  - Smoke testing
  - Post-deployment testing
  - Hot fixes

4.2 Explain concepts related to application service levels. (2)

- Uptime
- Service-level objectives (SLOs)
- Service-level agreements (SLAs)
- Mean time to repair (MTTR)
- Mean time between failures (MTBF)
- Feedback loop

4.3 Compare and contrast different methods to secure a connection to providers.

- Command-line interface (CLI) configuration
- Software development kit (SDK)
- Identity access management (IAM)
  - Machine identities
- Accessing external APIs