? EASY (10 Questions)

Q1. Infrastructure as Code (IaC) is primarily used to:

A. Monitor network traffic
B. Automate and manage infrastructure using code
C. Create Docker containers
D. Execute SQL queries

Q2. Terraform is classified as a:

A. Container orchestration tool
B. Version control system
C. Declarative IaC tool
D. Continuous integration tool

Q3. The main configuration file for Terraform typically has the extension:

A. .yml
B. .tf
C. .json
D. .conf

Q4. Terraform works with which concept to define cloud resources?

A. Imperative scripting
B. Declarative configuration
C. Manual provisioning
D. Continuous integration

Q5. A Terraform provider is used to:

A. Specify which cloud or service to manage
B. Define network subnets only
C. Run Ansible playbooks
D. Deploy Docker containers

Q6. Terraform `terraform init` command:

A. Applies infrastructure changes
B. Initializes the working directory and downloads providers
C. Deletes state files
D. Builds Docker images

Q7. Terraform `terraform plan` is used to:

A. Apply changes directly to cloud
B. Show proposed changes without applying
C. Delete all infrastructure
D. Launch EC2 instances

Q8. Terraform state files are stored in:

A. .tf files
B. terraform.tfstate
C. YAML files
D. JSON only

Q9. Terraform modules are used for:

A. Automating CI/CD pipelines
B. Reusing and organizing infrastructure code
C. Building containers
D. Monitoring applications

Q10. A key advantage of Terraform is:

A. Manual provisioning of servers
B. Infrastructure versioning and reproducibility
C. Container orchestration
D. Code compilation

□ MEDIUM (15 Questions)

Q11. Terraform uses which language to define configurations?

A. HCL (HashiCorp Configuration Language)
B. Python
C. YAML
D. JSON exclusively

Q12. Scenario: You need to create an AWS EC2 instance using Terraform. Which files are essential?

A. main.tf, provider.tf, variables.tf
B. docker-compose.yml
C. playbook.yml
D. state.tf

Q13. Terraform `terraform apply` command:

A. Shows proposed changes
B. Applies configuration changes to the infrastructure
C. Deletes local files only
D. Initializes provider plugins

Q14. Terraform remote backend is used to:

A. Store modules only
B. Manage state files centrally for collaboration
C. Launch containers
D. Run Ansible tasks

Q15. Terraform variables.tf file is used to:

A. Define reusable variable values for configuration
B. Store state files
C. Launch Docker containers
D. Create IAM users

Q16. Scenario: Multiple developers are working on the same environment. Which approach prevents state conflicts?

A. Local state files
B. Remote backend with locking (S3 + DynamoDB)
C. YAML files
D. Docker volumes

Q17. Terraform outputs are used to:

A. Display resource attributes after apply
B. Initialize provider
C. Delete infrastructure
D. Manage Ansible roles

Q18. Terraform `terraform destroy` command:

A. Shows planned changes
B. Removes all resources defined in the configuration
C. Updates state only
D. Runs modules

Q19. Scenario: You need to create reusable infrastructure code for VPC, subnets, and security groups. What Terraform feature is ideal?

A. Resource blocks only
B. Modules
C. Variables.tf only
D. Outputs.tf only

Q20. Terraform interpolation syntax uses:

A. ${}
B. []
C. ()
D. ##

Q21. Terraform `terraform validate` is used to:

A. Apply changes
B. Check syntax and configuration validity
C. Destroy resources
D. Build Docker images

Q22. Terraform resource block defines:

A. Configuration of a single infrastructure object
B. State backend
C. Module inputs
D. Provider authentication

Q23. Scenario: You need to pass sensitive credentials to Terraform. Which method is recommended?

A. Hardcode in main.tf
B. Use environment variables or Terraform Vault integration
C. Store in outputs.tf
D. Docker volumes

Q24. Terraform lifecycle meta-argument `create_before_destroy` ensures:

A. Resources are destroyed before creation
B. New resources are created before old ones are destroyed
C. Outputs are displayed first
D. Remote state is locked

Q25. Terraform `terraform fmt` command:

A. Validates configuration
B. Formats configuration files consistently
C. Destroys resources
D. Initializes backend

🟢 HARD (15 Questions)

Q26. Scenario: You want to organize multiple environments (dev, staging, prod) using Terraform. Best practice:

A. Single state file for all environments
B. Separate workspaces or state files per environment
C. Docker volumes
D. YAML inventory

Q27. Terraform `count` argument allows:

A. Reusable module creation
B. Creating multiple instances of a resource dynamically
C. Remote state management
D. Output formatting

Q28. Terraform `for_each` is used to:

A. Loop over a map or set of resources
B. Destroy resources
C. Initialize modules
D. Format code

Q29. Scenario: You want a module to be reusable across multiple projects. Which practice is recommended?

A. Hardcode resource names
B. Use input variables and outputs
C. Avoid modules
D. Store state locally only

Q30. Terraform backend types include:

A. Local, S3, Consul, Terraform Cloud
B. Docker, Kubernetes
C. Ansible, Puppet
D. YAML, JSON

Q31. Scenario: You update a Terraform configuration and want to preview changes without applying. Command:

A. terraform init
B. terraform plan
C. terraform apply
D. terraform destroy

Q32. Terraform module registry is used to:

A. Store Docker images
B. Discover and reuse community or private modules
C. Run Ansible playbooks
D. Monitor resources

Q33. Terraform `depends_on` argument:

A. Specifies explicit resource dependencies
B. Deletes modules automatically
C. Creates variables
D. Initializes providers

Q34. Scenario: You accidentally modified a live resource outside Terraform. Which command can help reconcile state?

A. terraform refresh
B. terraform plan
C. terraform apply
D. terraform init

Q35. Terraform `provider` block must include:

A. Resource type only
B. Authentication credentials and endpoint
C. Output variables
D. Module inputs

Q36. Terraform workspaces allow:

A. Managing multiple state files for the same configuration
B. Creating Kubernetes pods
C. Running Docker containers
D. Defining Ansible roles

Q37. Scenario: You want to create a VPC module with subnet and security group variables. Which is best practice?

A. Use hardcoded resource names
B. Define input variables for names, CIDRs, and security rules
C. Avoid modules
D. Store outputs only

Q38. Terraform `lifecycle` block `prevent_destroy` is used to:

A. Allow resource deletion
B. Prevent accidental deletion of critical resources
C. Initialize provider
D. Format configuration files

Q39. Scenario: You want to store Terraform state securely in AWS and enable locking. Which combination is correct?

A. S3 bucket + DynamoDB table
B. Local tfstate only
C. YAML file
D. Docker volume

Q40. Terraform output values are primarily used to:

A. Store state
B. Share information about resources after apply
C. Initialize backend
D. Loop over resources