Question 1:

Which one of the following command will rewrite Terraform configuration files to a canonical format and style.

- A. terraform fmt **(Correct)**
- B. terraform graph
- C. terraform graph -h
- D. terraform init

Question 2:

terraform refresh command will not modify infrastructure, but does modify the state file.

- TRUE **(Correct)**
- FALSE

Explanation

This is true for terraform refresh.

Question 3:

Which one of the following will run echo 0 and echo 1 on a newly created host

- A. provisioner "remote-exec" {
inline = [
"echo 0",
"echo 1"
]
}**(Correct)**

- B. provisioner "remote-exec" {
inline = [
echo 0,
echo 1
]
}

- C. provisioner "local-exec" { command = "echo 0"
command = "echo 1"
}

- D. provisioner "remote-exec" {
command = "${echo 0}"
command = "${echo 1}"
}

Question 4:

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or all of the arguments are omitted, we call this a _____.

A. incomplete configuration

B. partial configuration **(Correct)**

C. default configuration

D. changing configuration

E. first time configuration

Explanation

https://www.terraform.io/docs/backends/config.html

Question 5:

You want to get involved in the development of Terraform. As this is an open source project, you would like to contribute a fix for an open issue of Terraform. What programming language will need to use to write the fix?

A. Go

(Correct)

B. Python

C. It depends on which command the issue relates to

D. Java

Explanation

Terraform is written in the Go programming language. Terraform providers are also written in the Go programming language, although it is technically possible to use other languages for Terraform providers.

Question 6: Correct

You are using a terraform operation that writes state. Unfortunately automatic state unlocking has failed for that operation. Which of the below commands can be used to remove the already acquired lock on the state ?

A. terraform state unlock

B. terraform force-unlock

(Correct)

C. terraform unlock

D. None of the above

Explanation

https://www.terraform.io/docs/commands/force-unlock.html

https://www.terraform.io/docs/state/locking.html

Terraform has a force-unlock command to manually unlock the state if unlocking failed.

Be very careful with this command. If you unlock the state when someone else is holding the lock it could cause multiple writers. Force unlock should only be used to unlock your own lock in the situation where automatic unlocking failed.

Question 7: Correct

Which of the below features of Terraform can be used for managing small differences between different environments which can act more like completely separate working directories.

A. Repositories

B. Environment Variables

C. Backends

D. Workspaces

(Correct)

Explanation

https://www.terraform.io/docs/state/workspaces.html

Question 8: Correct

Which of the below configuration file formats are supported by Terraform? (Select TWO)

A. YAML

B. HCL

(Correct)

C. JSON

(Correct)

D. Node

E. Go

Explanation

Terraform supports both HashiCorp Configuration Language (HCL) and JSON formats for configurations.
https://www.terraform.io/docs/configuration/

Question 9: Correct

You are reviewing Terraform configurations for a big project in your company. You noticed that there are several identical sets of resources that appear in multiple configurations. What feature of Terraform would you recommend to use to reduce the amount of cloned configuration between the different configurations?

A. Packages

B. Modules

(Correct)

C. Provisioners

D. Backends

Explanation

Modules are reusable configuration packages that Terraform can share through a variety of sources including Terraform Registries, GitHub, and Amazon S3 buckets.
https://www.terraform.io/docs/modules/index.html
Question 10: Correct
What is the default backend for Terraform?

A. Default

B. Consul

C. Local
(Correct)

D. S3
Explanation
By default, Terraform uses the "local" backend. You must configure a backend block in order to use any other type of backend.
https://www.terraform.io/docs/backends/index.html
Question 11: Correct
`terraform refresh` **will update the state file?**

TRUE
(Correct)

FALSE
Explanation
The `terraform refresh` command is used to reconcile the state Terraform knows about (via its state file) with the real-world infrastructure. This can be used to detect any drift from the last-known state, and to update the state file.
This does not modify infrastructure, but does modify the state file. If the state is changed, this may cause changes to occur during the next plan or apply.
Question 12: Correct
You want terraform plan and apply to be executed in Terraform Cloud's run environment but the output is to be streamed locally. Which one of the below you will choose ?

A. This can be done using any of the local or remote backends

B. Remote Backends
(Correct)

C. Local Backends

D. Terraform Backends
Explanation
The remote backend stores Terraform state and may be used to run operations in Terraform Cloud.

When using full remote operations, operations like `terraform plan` or `terraform apply` can be executed in Terraform Cloud's run environment, with log output streaming to the local terminal.
https://www.terraform.io/docs/backends/types/remote.html

Question 13: Correct

Refer to the following terraform variable definition

```
variable "track_tag" {
  type = list
  default = ["data_ec2","integration_ec2","digital_ec2"]
}

track_tag = {
    Name = element(var.track_tag,count.index)
}
```

If count.index is set to 2, which of the following values will be assigned to the name attribute of track_tag variable?

A. integration_ec2

B. data_ec2

C. digital_ec2

(Correct)

D. track_tag

Question 14: Correct

What allows you to conveniently switch between multiple instances of a single configuration within its single backend?

A. Local Backends

B. Workspaces

(Correct)

C. Remote Backends

D.Providers

Question 15: Correct

A single terraform resource file that defines an aws_instance resource can simply be renamed to vsphere_virtual_machine in order to switch cloud providers.

TRUE

FALSE

(Correct)

Explanation

Every provider has its own required and allowed declarations none of which match between cloud providers.

Question 16: Correct

You have multiple developers working on a terraform project (using terraform OSS), and have saved the terraform state in a remote S3 bucket . However ,team is intermittently experiencing inconsistencies in the provisioned infrastructure / failure in the code . You have traced this problem to simultaneous/concurrent runs of terraform apply command for 2/more developers . What can you do to fix this problem ?

A. Structure your team in such a way that only one individual will run terraform apply , everyone will just make changes and share with him. Then there will be no chance of any inconsistencies.

B. Stop using remote state , and store the developer tfstate in their own machine . Once a day , all developers should sit together and merge the state files manually , to avoid any inconsistencies.

C. Use terraform workspaces feature, this will fix this problem by default , as every developer will have their own state file , and terraform will merge them on server side on its own.

D. Enable terraform state locking for the S3 backend using DynamoDB table. This prevents others from acquiring the lock and potentially corrupting your state.

(Correct)

Explanation

S3 backend support state locking using DynamoDB.

https://www.terraform.io/docs/state/locking.html

Question 17: Correct

When using Terraform in a team it is important for everyone to be working with the same state so that operations will be applied to the same remote objects. Which of the below option is a recommended solution for this?

A. Workspace

B. Remote State

(Correct)

C. Use the cached state and treat this as the record of truth.

D. Module

Explanation

https://www.terraform.io/docs/state/remote.html

Question 18: Correct

Which of the following Terraform files should be ignored by Git when committing code to a repo? (select Three)

A. terraform.tfstate

(Correct)

B. input.tf

C. output.tf

D. Any files with names ending in `.auto.tfvars` or `.auto.tfvars.json`.

(Correct)

E. Files named exactly `terraform.tfvars` or `terraform.tfvars.json`.

(Correct)

Explanation

The .gitignore file should be configured to ignore Terraform files that either contain sensitive data or aren't required to save.

The terraform.tfstate file contains the terraform state of a specific environment and doesn't need to be preserved in a repo. The terraform.tfvars file may contain sensitive data, such as passwords or IP addresses of an environment that you may not want to share with others.

Question 19: Correct

Which of the below commands will rename a EC2 instance without destroying and recreating it ?

A. terraform plan

B. terraform mv

C. terraform plan mv

D. terraform state mv

(Correct)

Question 20: Correct

Workspaces in Terraform provides similar functionality in the open-source, Terraform Cloud, and Enterprise versions of Terraform.

TRUE

FALSE

(Correct)

Explanation

https://www.terraform.io/docs/cloud/migrate/workspaces.html

Workspaces, managed with the `terraform workspace` command, aren't the same thing as Terraform Cloud's workspaces. Terraform Cloud workspaces act more like completely separate working directories; CLI workspaces are just alternate state files.

Question 21: Correct

Which of the following command can be used to view the specified version constraints for all providers used in the current configuration.

A. terraform providers

(Correct)

B. terraform provider

C. terraform state show

D. terraform plan

Explanation

Use the terraform providers command to view the specified version constraints for all providers used in the current configuration.

https://www.terraform.io/docs/configuration/providers.html

Question 22: Correct

You have created an AWS EC2 instance of type t2.micro through your terraform configuration file ec2.tf . Now you want to change the instance type from t2.micro to t2.medium. Accordingly you have changed your configuration file and and ran terraform plan. After running terraform plan you check the output and saw one instance will be updated from t2.micro --> t2.medium. After this you went to grab a coffee without running terraform apply and meanwhile a member of your team changed the instance type of that EC2 instance to t2.medium from aws console. After coming to your desk you run terraform apply. What will happen ?

A. terraform apply will through an error.

B. The instance type will be changed to t2.micro and again will be changed to t2.medium

C. No resource will be updated and you will see the message : Apply Complete ! Resources : 0 added, 0 changed, 0 destroyed.

(Correct)

D. 1 resource will be updated and you will see the message : Apply Complete ! Resources : 0 added, 1 changed, 0 destroyed.

Explanation

C is the correct ans. You may see same kind of questions with different resource type in exam.

Question 23: Correct

Which of the below terraform commands do not run terraform refresh implicitly before taking actual action of the command ?

A. terraform init

(Correct)

B. terraform plan

C. terraform apply

D. terraform destroy

☑
E. terraform import
(Correct)
Explanation
You may see the same question in the actual exam with less options. Terraform init and import are the correct answer. Though in case of terraform import the state refresh is done after the import completes. Check the available command line flag `-refresh=true` for all these commands for more detail .

```
$ terraform import aws_instance.import_example i-03efafa258104165f
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...
aws_instance.import_example: Import complete!
  Imported aws_instance (ID: i-03efafa258104165f)
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f)

Import successful!

The resources that were imported are shown above. These resources are now in
your Terraform state and will henceforth be managed by Terraform.
```

Question 24: Correct
Terraform works well in Windows but a Windows server is required.
◌
TRUE
◉
FALSE
(Correct)
Explanation
You may see this question in actual exam. Please remember : Terraform does not require GO language to be installed as a prerequisite and it does not require a Windows Server as well.

Question 25: Correct
You have provisioned some aws resources in your test environment through Terraform for a POC work. After the POC, now you want to destroy the resources but before destroying them you want to check what resources will be getting destroyed through terraform. what are the options of doing that? (Select TWO)
☐
A. This is not possible
☑
B. Use terraform destroy command
(Correct)
☑
C. Use `terraform plan -destroy` command.
(Correct)
☐
D. Use terraform plan command
Explanation

The behavior of any `terraform destroy` command can be previewed at any time with an equivalent `terraform plan -destroy` command

terraform destroy command also shows what will be destroyed and assk for confirmation.
Terraform plan is not applicable here to show you the execution plan as you have not deleted the resources from your configuration file. This is important for the exam.

Question 26: Correct

Which of the below datatype is not supported by Terraform.

A. List

B. Map

C. Object

D. Array

(Correct)

Explanation

Array is not a supported collection datatype of terraform. This is important for exam.

Question 27: Correct

If you enable TF_LOG = DEBUG, the log will be stored in syslog.log file in the currect directory

A. TRUE

B. FALSE

(Correct)

Explanation

https://www.terraform.io/docs/internals/debugging.html. This is important for exam

Question 28: Correct

While using generic git repository as a module source, which of the below options allows terraform to select a specific version or tag instead of selecting the HEAD.

A. Append ref argument as

```
module "vpc" {
  source = "git::https://example.com/vpc.git?ref=v1.2.0"
}
```

(Correct)

B. Append ref argument as

```
module "vpc" {
  source = "git::https://example.com/vpc.git#ref=v1.2.0"
}
```

C. By default, Terraform will clone and use the default branch (referenced by `HEAD`) in the selected repository and you can not override this.

D. Append version argument as

```
module "vpc" {
  source = "git::https://example.com/vpc.git?version=v1.2.0"
}
```

Explanation

This is important for exam. Please refer https://www.terraform.io/docs/modules/sources.html

Question 29: Correct

You have created 2 workspaces PROD and RQA. You have switched to RQA and provisioned RQA infrastructure from this workspace. Where is your state file stored?

A. .terraform.d

B. terraform.tfstate

C. terraform.tfstate.RQA

D. terraform.tfstate.d

(Correct)

Question 30: Correct

`lookup` **retrieves the value of a single element from which of the below data type ?**

A. List

B. Map

(Correct)

C. Set

D. String

Explanation

General syntax of lookup function is

```
lookup(map, key, default)
```

Question 31: Correct

John wants to use two different regions to deploy two different EC2 instances. He has specified two provider blocks in his providers.tf file.

provider "aws" { region = "us-east-1" }

provider "aws" { region = "us-west-2" }

When he run terraform plan he encountered an error. How to fix this?

A. It can not be fixed

B. Use alias for region = "us-west-2"

(Correct)

C. Use default keyword with region = "us-east-1"

D. Use another provider version

Question 32: Correct

Matt wants to import a manually created EC2 instance into terraform so that he can manage the EC2 instance through terraform going forward. He has written the configuration file of the EC2 instance before importing it to Terraform. Following is the code:

```
resource "aws_instance" "matt_ec2" {
 ami = "ami-bg2640de"
 instance_type = "t2.micro"
 vpc_security_group_ids = ["sg-6ae7d613", "sg-53370035"]
 key_name = "mysecret"
 subnet_id = "subnet-9e3cfbc5"
}
```

The instance id of that EC2 instance is i-0260835eb7e9bd40 How he can import data of EC2 to state file?

A. terraform import aws_instance.i-0260835eb7e9bd40

B. terraform import aws_instance.matt_ec2 i-0260835eb7e9bd40

(Correct)

C. terraform import aws_instance.id = i-0260835eb7e9bd40

D terraform import i-0260835eb7e9bd40

Explanation

https://www.terraform.io/docs/import/usage.html

Question 33: Correct

By default, a defined provisioner is a creation-time provisioner.

TRUE

(Correct)

FALSE

Explanation

You must explicitly define a provisioner to be a destroy-time provisioner

Creation-Time Provisioners

By default, provisioners run when the resource they are defined within is created. Creation-time provisioners are only run during *creation*, not during updating or any other lifecycle. They are meant as a means to perform bootstrapping of a system.

If a creation-time provisioner fails, the resource is marked as **tainted**. A tainted resource will be planned for destruction and recreation upon the next `terraform apply`. Terraform does this because a failed provisioner can leave a resource in a semi-configured state. Because Terraform cannot reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

You can change this behavior by setting the `on_failure` attribute, which is covered in detail below.

»Destroy-Time Provisioners

If `when = "destroy"` is specified, the provisioner will run when the resource it is defined within is *destroyed*.

```
resource "aws_instance" "web" {
  # ...

  provisioner "local-exec" {
    when    = "destroy"
    command = "echo 'Destroy-time provisioner'"
  }
}
```

Destroy provisioners are run before the resource is destroyed. If they fail, Terraform will error and rerun the provisioners again on the next `terraform apply`. Due to this behavior, care should be taken for destroy provisioners to be safe to run multiple times.

Destroy-time provisioners can only run if they remain in the configuration at the time a resource is destroyed. If a resource block with a destroy-time provisioner is removed entirely from the configuration, its provisioner configurations are removed along with it and thus the destroy provisioner won't run. To work around this, a multi-step process can be used to safely remove a resource with a destroy-time provisioner:

Update the resource configuration to include `count = 0`.

Apply the configuration to destroy any existing instances of the resource, including running the destroy provisioner.

Remove the resource block entirely from configuration, along with its `provisioner` blocks.

Apply again, at which point no further action should be taken since the resources were already destroyed.

This limitation may be addressed in future versions of Terraform. For now, destroy-time provisioners must be used sparingly and with care.

https://www.terraform.io/docs/provisioners/index.html

Question 34: Correct

Your company has been using Terraform Cloud for a some time now . But every team is creating their own modules , and there is no standardization of the modules , with each team creating the resources in their own unique way . You want to enforce a standardization of the modules across the enterprise . What should be your approach.

A. Upgrade to Terraform enterprise , since this is not possible in terraform cloud.

B. Implement a Private module registry in Terraform cloud , and ask teams to reference them. **(Correct)**

C. Upload the modules in the terraform public module registry , and ask teams to reference them

D. Create individual workspaces for each team , and ask them to share modules across workspaces.

Explanation

Understand the different offerings in Terraform OS, Terraform Cloud and Terraform Enterprise.

Terraform Cloud's private module registry helps you share Terraform modules across your organization.

https://www.terraform.io/docs/cloud/registry/index.html

Question 35: Correct

```
resource "aws_s3_bucket" "example" {
   bucket = "my-test-s3-terraform-bucket"
…}
resource "aws_iam_role" "test_role" {
    name = "test_role"
…}
```

Due to the way that the application code is written , the s3 bucket must be created before the test role is created , otherwise there will be a problem. How can you ensure that?

A. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.

B. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.

C. Add explicit dependency using depends_on . This will ensure the correct order of resource creation.
(Correct)

D. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.

Explanation

Implicit dependency works only if there is some reference of one resource to another. Explicit dependency is the option here.

Question 36: Correct

A data block requests that Terraform read from a given data source and export the result under the given local name.

TRUE
(Correct)

FALSE

Question 37: Correct

Which of the following type of variable allows multiple values of several distinct types to be grouped together as a single value?

A. List

B. Object
(Correct)

C. Tuple
(Correct)

D. Map

Explanation
Structural type of variable allows multiple values of several distinct types to be grouped together as a single value. They require a schema as an argument, to specify which types are allowed for which elements.
https://www.terraform.io/docs/configuration/types.html
Question 38: Correct
When using remote state, state is only ever held in memory when used by Terraform.
◉
TRUE
(Correct)
○
FALSE
Question 39: Correct
Please identify the offerings which are unique to Terraform Enterprise , and not available in either Terraform OSS , or Terraform Cloud. Select four.
☑
A. SAML/SSO
(Correct)
☑
B. Audit Logs
(Correct)
☐
C. Sentinel
☑
D. Private Network Connectivity
(Correct)
☑
E. Clustering
(Correct)
☐
F. VCS Integration
Explanation
Sentinel, VCS Integration are offered in Terraform Cloud. Everything that is in Terraform cloud is already included in Terraform Enterprise. But A, B ,D , E are unique features of Terraform Enterprise. Please read compare offering section. This can help you score 2-3 questions in the exam.
https://www.hashicorp.com/products/terraform/pricing/
Question 40: Correct
How can you ensure that the engineering team who has access to git repo will not create any non-compliant resources that might lead to a security audit failure in future. your team is using Hashicorp Terraform Enterprise Edition.
○
A. Implement a review process where every code will be reviewed before merging to the master branch.
○
B. Create a design /security document (in PDF) and share to the team , and ask them to always follow that document , and never deviate from it.
◉

C. Since your team is using Hashicorp Terraform Enterprise Edition , enable Sentinel , and write Policy-As-Code rules that will check for non-compliant resource provisioning , and prevent/report them
(Correct)

D. Use Terraform OSS Sentinel Lite version , which will save cost , since there is no charge for OSS , but it can still check for most non-compliant rules using Policy-As-Code

Explanation

https://www.terraform.io/docs/cloud/sentinel/index.html

Question 41: Correct

Which of the below are paid features of Terraform Cloud?

A. Sentinel policies
(Correct)

B. Full API Coverage

C. Private Module Registry

D. Roles/ Team management
(Correct)

E. Cost Estimation
(Correct)

F. Secure variable Storage

Explanation

B, C and F are part of free tier of Terraform Cloud. Please review the below link properly. You can get couple of questions from the below comparison. Learn different offering between Terraform open source, Terraform Cloud(different tiers) and Terraform Enterprise

https://www.hashicorp.com/products/terraform/pricing/

Question 42: Correct

The `terraform init` command is always safe to run multiple times, to bring the working directory up to date with changes in the configuration. Though subsequent runs may give errors, this command will never delete your existing configuration or state.

TRUE
(Correct)

FALSE

Explanation

https://www.terraform.io/docs/commands/init.html

Question 43: Correct

What does terrafom plan do ?

A. Create an execution plan by evaluating the difference between configuration file and state file.
(Correct)

B. Create an execution plan by evaluating the difference between configuration file and actual infrastructure.

C. Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.

D. Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.

Explanation

This is important for the exam. Please review all the definition of terraform plan, apply, destroy command.