TASKS ON TERRAFORM

- **NAME: S.MUZZAMMIL HUSSAIN**
- **DATE: 23/07/2025**
- **BATCH: 11**
- **NO.OF TASKS: 1**

Task1. Provide all data types to random providers

Create a file named **variables.tf** using **vi variables.tf** and enter the following and save using esc:wq

```
# Primitive types
variable "my_string" {
  description = "Example string"
  type        = string
}
variable "my_number" {
  description = "Example number"
  type        = number
}
variable "my_bool" {
  description = "Example boolean"
  type        = bool
}
# Complex types
variable "my_list" {
  description = "List of strings"
  type        = list(string)
}
variable "my_set" {
  description = "Set of numbers"
  type        = set(number)
}
```

```
variable "my_map" {
  description = "Map of string values"
  type        = map(string)
}
variable "my_object" {
  description = "Object with mixed types"
  type = object({
    id      = string
    enabled = bool
    score   = number
  })
}
variable "my_tuple" {
  description = "Tuple with mixed types"
  type        = tuple([string, number, bool])
}
```

Create a file named **main.tf** using **vi main.tf** and enter the following and save using esc:wq

```
provider "random" {}
resource "random_string" "string" {
  length  = var.my_number
  special = var.my_bool
}
resource "random_id" "id" {
  byte_length = var.my_number
}
resource "random_pet" "list_pet" {
  for_each = toset(var.my_list)
}
resource "random_shuffle" "shuffle_set" {
  input        = tolist(var.my_set)
  result_count = length(var.my_set)
}
resource "random_string" "map_example" {
  for_each = var.my_map
  length   = 8
  special  = false
}
output "object_output" {
  value = var.my_object
}
output "tuple_output" {
  value = var.my_tuple
}
```

```
provider "random" {}

resource "random_string" "string" {
  length  = var.my_number
  special = var.my_bool
}

resource "random_id" "id" {
  byte_length = var.my_number
}

resource "random_pet" "list_pet" {
  for_each = toset(var.my_list)
}

resource "random_shuffle" "shuffle_set" {
  input        = tolist(var.my_set)
  result_count = length(var.my_set)
}

resource "random_string" "map_example" {
  for_each = var.my_map
  length   = 8
  special  = false
}

output "object_output" {
  value = var.my_object
}

output "tuple_output" {
  value = var.my_tuple
}
```

Create a file named **terraform.tfvars** using **vi terraform.tfvars** and enter the following and save using esc:wq

my_string = "production"

my_number = 6

my_bool = true

my_list = ["red", "green", "blue"]

my_set = [2, 4, 6]

my_map = {

env = "prod"

tier = "web"

}

my_object = {

id = "obj-123"

enabled = false

score = 42

}

my_tuple = ["data", 100, true]

```
my_string = "production"
my_number = 6
my_bool   = true

my_list = ["red", "green", "blue"]

my_set  = [2, 4, 6]

my_map = {
  env  = "prod"
  tier = "web"
}

my_object = {
  id      = "obj-123"
  enabled = false
  score   = 42
}

my_tuple = ["data", 100, true]
~
~
```

Run the command **terraform init** and **terraform validate**

```
mujju@VMterra:~/b11/2307$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/random...
- Installing hashicorp/random v3.7.2...
- Installed hashicorp/random v3.7.2 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
mujju@VMterra:~/b11/2307$ terraform validate
Success! The configuration is valid.
```

Run the command **terraform apply**

```
mujju@VMterra:~/b11/2307$ terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # random_id.id will be created
  + resource "random_id" "id" {
      + b64_std     = (known after apply)
      + b64_url     = (known after apply)
      + byte_length = 6
      + dec         = (known after apply)
      + hex         = (known after apply)
      + id          = (known after apply)
    }

  # random_pet.list_pet["blue"] will be created
  + resource "random_pet" "list_pet" {
      + id        = (known after apply)
      + length    = 2
      + separator = "-"
    }

  # random_pet.list_pet["green"] will be created
  + resource "random_pet" "list_pet" {
      + id        = (known after apply)
      + length    = 2
      + separator = "-"
    }

  # random_pet.list_pet["red"] will be created
  + resource "random_pet" "list_pet" {
      + id        = (known after apply)
      + length    = 2
      + separator = "-"
    }

  # random_shuffle.shuffle_set will be created
  + resource "random_shuffle" "shuffle_set" {
      + id           = (known after apply)
      + input        = [
          + "2",
          + "4",
          + "6",
        ]
      + result       = (known after apply)
      + result_count = 3
```

```
  # random_string.map_example["env"] will be created
  + resource "random_string" "map_example" {
      + id          = (known after apply)
      + length      = 8
      + lower       = true
      + min_lower   = 0
      + min_numeric = 0
      + min_special = 0
      + min_upper   = 0
      + number      = true
      + numeric     = true
      + result      = (known after apply)
      + special     = false
      + upper       = true
    }

  # random_string.map_example["tier"] will be created
  + resource "random_string" "map_example" {
      + id          = (known after apply)
      + length      = 8
      + lower       = true
      + min_lower   = 0
      + min_numeric = 0
      + min_special = 0
      + min_upper   = 0
      + number      = true
      + numeric     = true
      + result      = (known after apply)
      + special     = false
      + upper       = true
    }

  # random_string.string will be created
  + resource "random_string" "string" {
      + id          = (known after apply)
      + length      = 6
      + lower       = true
      + min_lower   = 0
      + min_numeric = 0
      + min_special = 0
      + min_upper   = 0
      + number      = true
      + numeric     = true
      + result      = (known after apply)
      + special     = true
      + upper       = true
    }
```

```
Plan: 8 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + object_output = {
      + enabled = false
      + id      = "obj-123"
      + score   = 42
    }
  + tuple_output  = [
      + "data",
      + 100,
      + true,
    ]
```

```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

random_string.map_example["tier"]: Creating...
random_pet.list_pet["blue"]: Creating...
random_pet.list_pet["green"]: Creating...
random_string.map_example["env"]: Creating...
random_id.id: Creating...
random_string.string: Creating...
random_shuffle.shuffle_set: Creating...
random_pet.list_pet["red"]: Creating...
random_id.id: Creation complete after 1s [id=-mDjBBvC]
random_pet.list_pet["red"]: Creation complete after 1s [id=whole-ant]
random_string.string: Creation complete after 1s [id=Y9eJT_]
random_string.map_example["tier"]: Creation complete after 1s [id=p3lwf12N]
random_shuffle.shuffle_set: Creation complete after 1s [id=-]
random_pet.list_pet["blue"]: Creation complete after 1s [id=fun-humpback]
random_pet.list_pet["green"]: Creation complete after 1s [id=still-satyr]
random_string.map_example["env"]: Creation complete after 1s [id=LN4mVSIs]

Apply complete! Resources: 8 added, 0 changed, 0 destroyed.

Outputs:

object_output = {
  "enabled" = false
  "id" = "obj-123"
  "score" = 42
}
tuple_output = [
  "data",
  100,
  true,
]
```

List contents using **tree -a**

```
mujju@VMterra:~/b11/2307$ tree -a
.
├── .terraform
│   └── providers
│       └── registry.terraform.io
│           └── hashicorp
│               └── random
│                   └── 3.7.2
│                       └── linux_amd64
│                           ├── LICENSE.txt
│                           └── terraform-provider-random_v3.7.2_x5
├── .terraform.lock.hcl
├── main.tf
├── terraform.tfstate
├── terraform.tfvars
└── variables.tf

7 directories, 7 files
mujju@VMterra:~/b11/2307$
```