TASKS ON TERRAFORM

- **NAME: S.MUZZAMMIL HUSSAIN**
- **DATE: 25/07/2025**
- **BATCH: 11**
- **NO.OF TASKS: 1**

Task 1. Explain for expression in terraform

In Terraform, the **for expression** (often called a "for loop") is a way to iterate over collections like lists or maps to transform or filter their elements, producing new lists or maps as output. It is used inside expressions rather than as traditional imperative loops.

Basic Syntax

1. For **lists**:
[for <item> in <list> : <expression>]

This iterates over each item in the list and produces a new list where each element is the result of the <expression> applied on item.

2. For **maps**:
[for <key>, <value> in <map> : <expression>]
Similar to lists, but iterates over key-value pairs.

3. To produce a new **map** instead of a list:
{for <key>, <value> in <map> : <new_key> => <new_value>}
This creates a map with transformed keys and/or values.

Example:

Create a **main.tf** file using **vi main.tf** add the following configuration:

```
locals {

        filename_upper =[for value in var.filename: upper(value)]

        map_keys = [ for key, value in  var.filnamemap : upper(key) ]

        map_values = [ for key, value in  var.filnamemap : upper(value) ]

        map_upper = { for key, value in  var.filnamemap : key => upper(value) }

}



output a1{

         value = local.filename_upper

}

output a2{

         value = local.map_keys

}

output a3{
```

```
            value = local.map_values
}
output a4{
            value = local.map_upper
}
variable "filename" {
  type    = list(string)
  default = ["a", "b", "c"]
}
resource "local_file" "f8" {
  count    = length(local.filename_upper)
  filename = local.filename_upper[count.index]
  content  = "test"
}
variable filnamemap {
  type    = map(string)
  default = {
           name ="a"
           address = "b"
           }
}
```

Explanation of Each Block

locals block:

filename_upper: Takes the variable filename (list of strings) and creates a new list with each element uppercased.

map_keys: Extracts the keys from filnamemap, converts them to uppercase, and outputs a list.

map_values: Extracts and uppercases all values in filnamemap, outputs as a list.

map_upper: Builds a new map with the original keys but replaces the values with their uppercased versions.

output blocks:

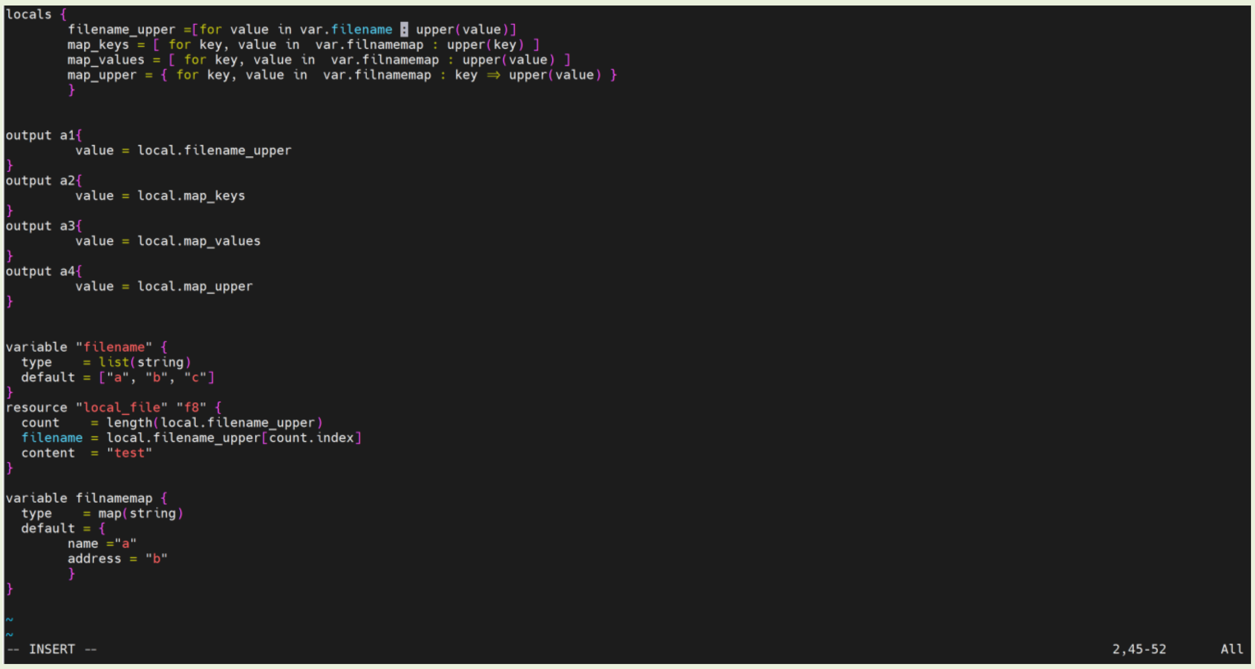
Print the results of each transformation to the console after running terraform apply.

variable blocks:

Define your input data: a list of filenames (["a", "b", "c"]) and a map with keys "name" and "address".

resource block:

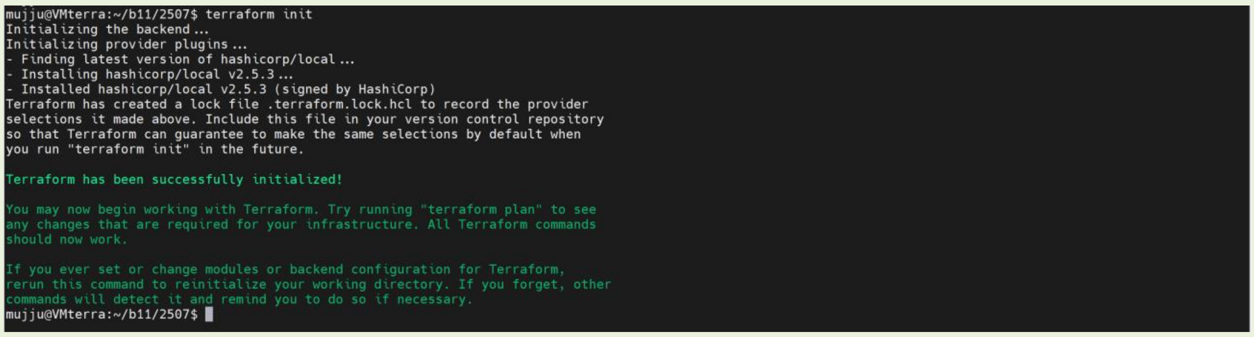
For each item in filename_upper, creates a local file named with the uppercase filename containing the text "test".



```
locals {
        filename_upper =[for value in var.filename : upper(value)]
        map_keys = [ for key, value in  var.filnamemap : upper(key) ]
        map_values = [ for key, value in  var.filnamemap : upper(value) ]
        map_upper = { for key, value in  var.filnamemap : key => upper(value) }
        }


output a1{
         value = local.filename_upper
}
output a2{
         value = local.map_keys
}
output a3{
         value = local.map_values
}
output a4{
         value = local.map_upper
}


variable "filename" {
  type    = list(string)
  default = ["a", "b", "c"]
}
resource "local_file" "f8" {
  count    = length(local.filename_upper)
  filename = local.filename_upper[count.index]
  content  = "test"
}

variable filnamemap {
  type    = map(string)
  default = {
        name ="a"
        address = "b"
        }
}

~
~
-- INSERT --                                                                          2,45-52       All
```

Initialize Terraform

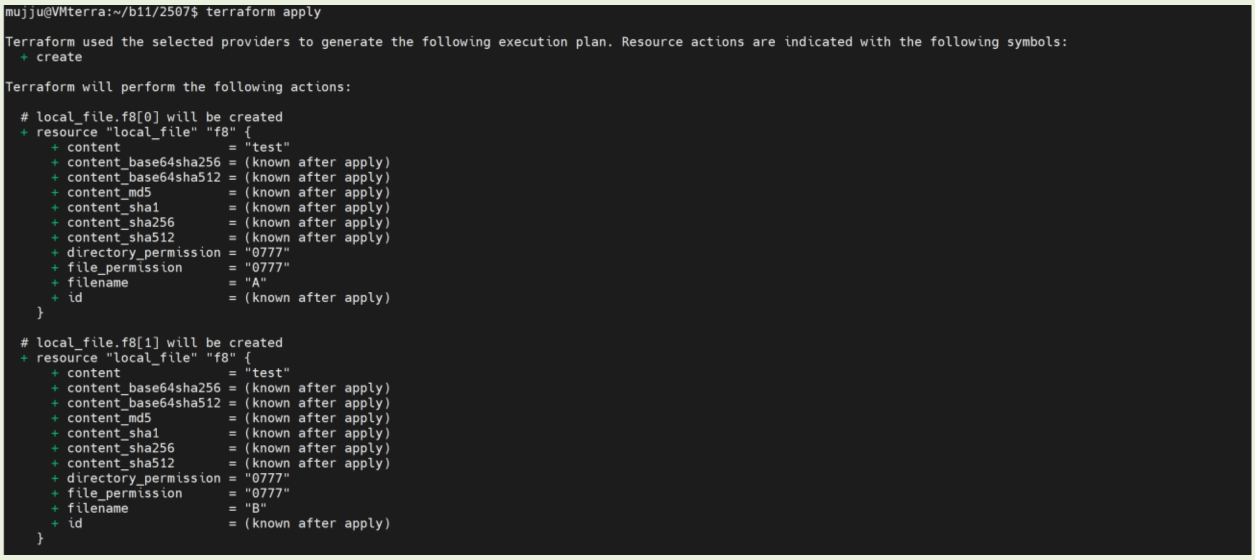
terraform init



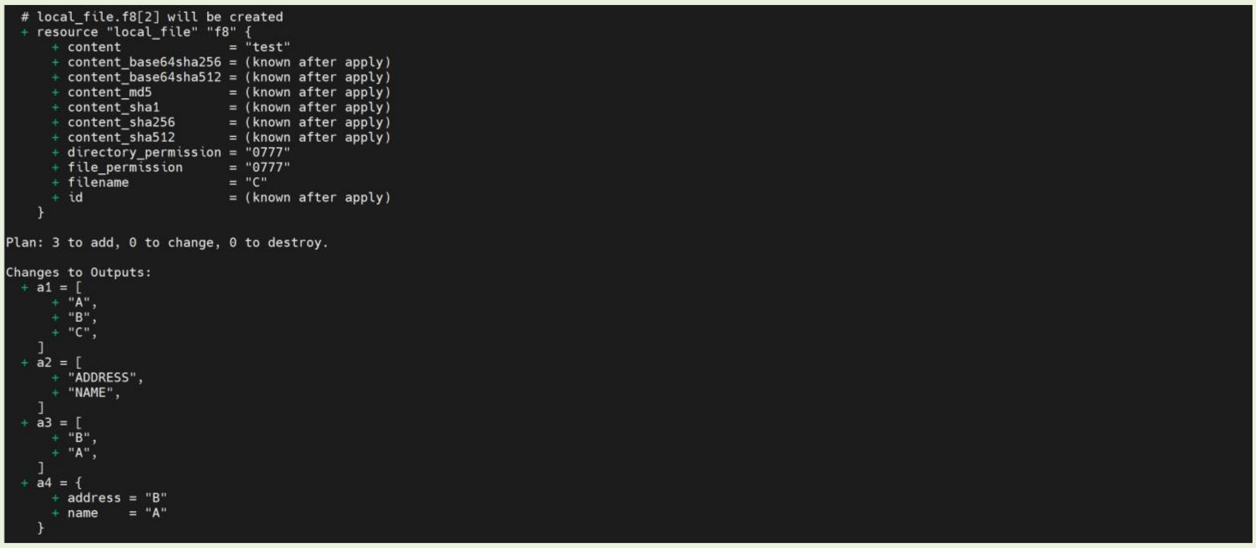
```
mujju@VMterra:~/b11/2507$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v2.5.3...
- Installed hashicorp/local v2.5.3 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl 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:~/b11/2507$
```

Run Terraform apply

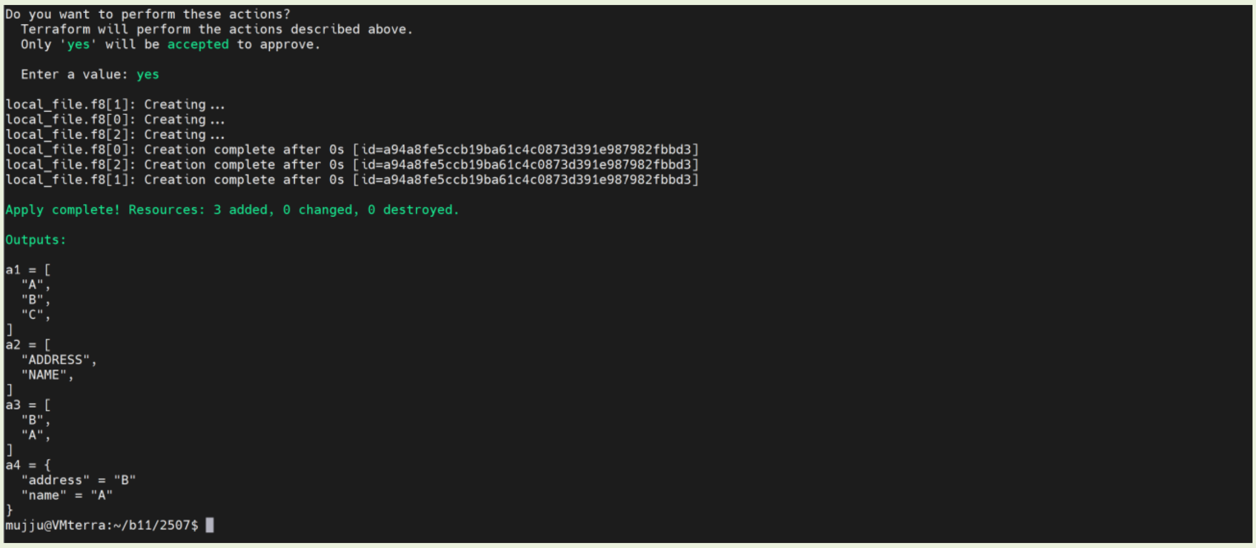
terraform apply



```
mujju@VMterra:~/b11/2507$ 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:

  # local_file.f8[0] will be created
  + resource "local_file" "f8" {
      + content              = "test"
      + content_base64sha256 = (known after apply)
      + content_base64sha512 = (known after apply)
      + content_md5          = (known after apply)
      + content_sha1         = (known after apply)
      + content_sha256       = (known after apply)
      + content_sha512       = (known after apply)
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "A"
      + id                   = (known after apply)
    }

  # local_file.f8[1] will be created
  + resource "local_file" "f8" {
      + content              = "test"
      + content_base64sha256 = (known after apply)
      + content_base64sha512 = (known after apply)
      + content_md5          = (known after apply)
      + content_sha1         = (known after apply)
      + content_sha256       = (known after apply)
      + content_sha512       = (known after apply)
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "B"
      + id                   = (known after apply)
    }
```

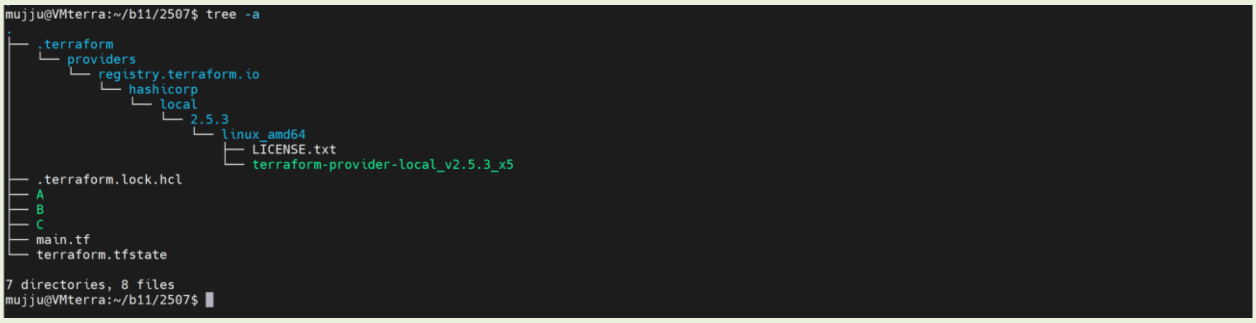


```
  # local_file.f8[2] will be created
  + resource "local_file" "f8" {
      + content              = "test"
      + content_base64sha256 = (known after apply)
      + content_base64sha512 = (known after apply)
      + content_md5          = (known after apply)
      + content_sha1         = (known after apply)
      + content_sha256       = (known after apply)
      + content_sha512       = (known after apply)
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "C"
      + id                   = (known after apply)
    }

Plan: 3 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + a1 = [
      + "A",
      + "B",
      + "C",
    ]
  + a2 = [
      + "ADDRESS",
      + "NAME",
    ]
  + a3 = [
      + "B",
      + "A",
    ]
  + a4 = {
      + address = "B"
      + name    = "A"
    }
```



```
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

local_file.f8[1]: Creating...
local_file.f8[0]: Creating...
local_file.f8[2]: Creating...
local_file.f8[0]: Creation complete after 0s [id=a94a8fe5ccb19ba61c4c0873d391e987982fbbd3]
local_file.f8[2]: Creation complete after 0s [id=a94a8fe5ccb19ba61c4c0873d391e987982fbbd3]
local_file.f8[1]: Creation complete after 0s [id=a94a8fe5ccb19ba61c4c0873d391e987982fbbd3]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

Outputs:

a1 = [
  "A",
  "B",
  "C",
]
a2 = [
  "ADDRESS",
  "NAME",
]
a3 = [
  "B",
  "A",
]
a4 = {
  "address" = "B"
  "name" = "A"
}
mujju@VMterra:~/b11/2507$
```

Check the contents using **tree -a**



```
mujju@VMterra:~/b11/2507$ tree -a
.
├── .terraform
│   └── providers
│       └── registry.terraform.io
│           └── hashicorp
│               └── local
│                   └── 2.5.3
│                       └── linux_amd64
│                           ├── LICENSE.txt
│                           └── terraform-provider-local_v2.5.3_x5
├── .terraform.lock.hcl
├── A
├── B
├── C
├── main.tf
└── terraform.tfstate

7 directories, 8 files
mujju@VMterra:~/b11/2507$
```