Terraform Task

**Name:Shaik Khaja Basha**

**Batch : Batch 11**

**Date : 25.07.2025**

**Task : for expression and functions in Terraform**

**Terraform for Expressions:**

Terraform's for expressions allow you to transform collections (lists, sets, maps) in powerful ways. Here's a comprehensive guide with examples:

**Basic Syntax**

**List Iteration**

[for item in LIST : EXPRESSION]

* Returns a new list
* item represents each element

**Map Iteration**

[for key, value in MAP : EXPRESSION]

{for key, value in MAP : KEY\_EXPRESSION => VALUE\_EXPRESSION}

* Can return either a list or new map

**Practical Examples**

**A. Transforming Lists**

variable "filename" {

type = list(string)

default = ["a", "b", "c"]

}

resource "local\_file" "f8" {

count = length(var.filename\_upper)

filename = var.filename\_upper[count.index]

content = "test"

}

**Working with Maps**

variable filnamemap {

type = map(string)

default = {

name ="a"

address = "b"

}

}

["name","address"]

["a",'b"]

**Create a file name for the configuration as res.tf and put the values save and quit ess:wq**

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" "f1" {

count = length(var.filename\_upper)

filename = var.filename\_upper[count.index]

content = "test"

}

variable filnamemap {

type = map(string)

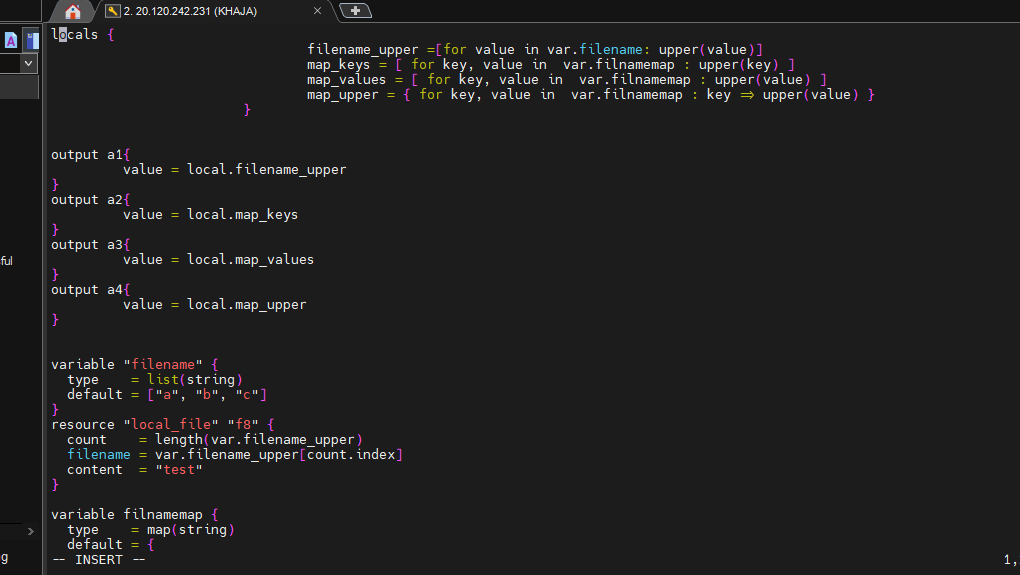
default = {

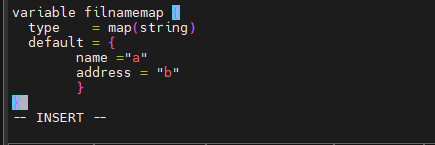
name ="a"

address = "b"

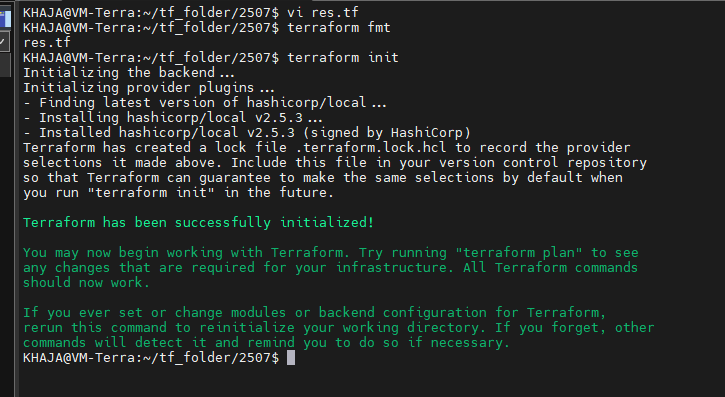
}

}

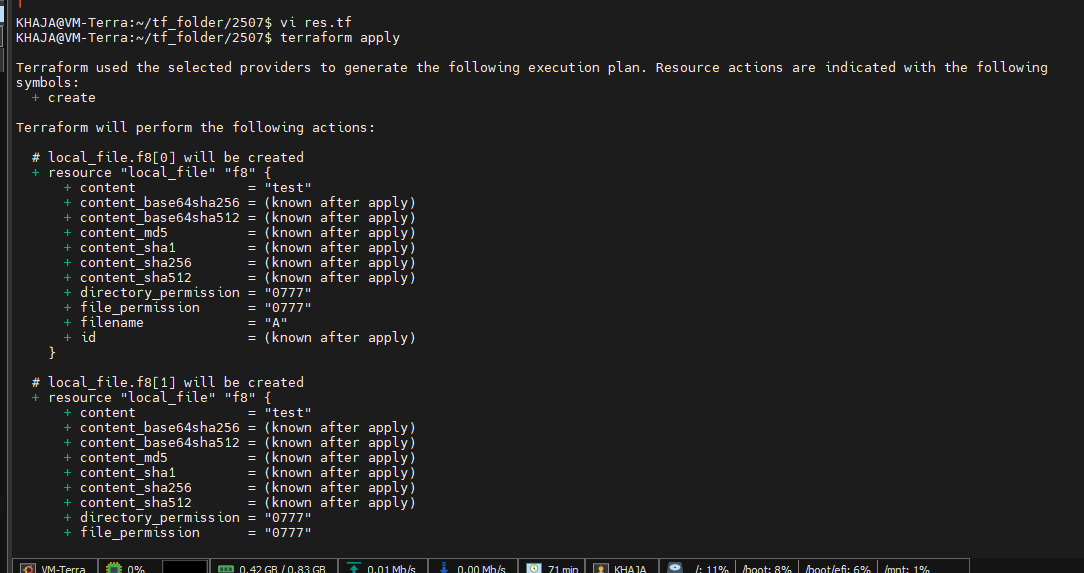


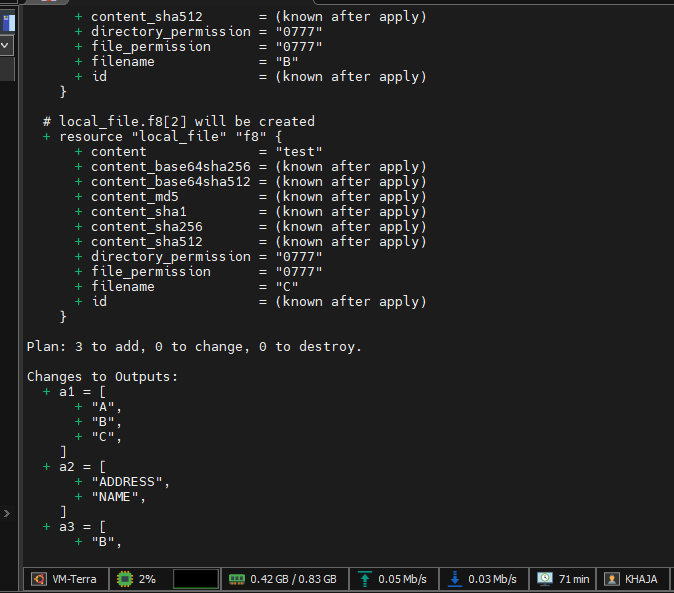


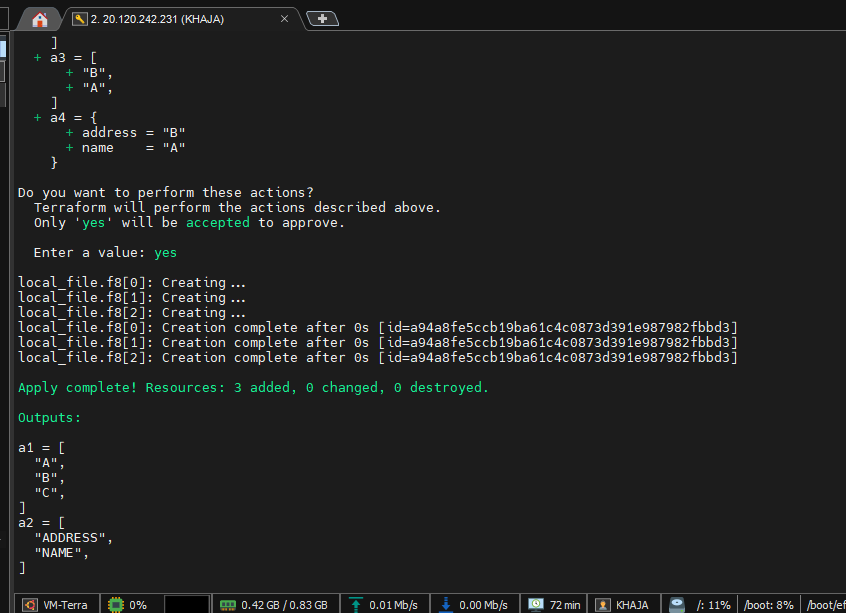
Execute the command **terraform init** for initializing the configuration

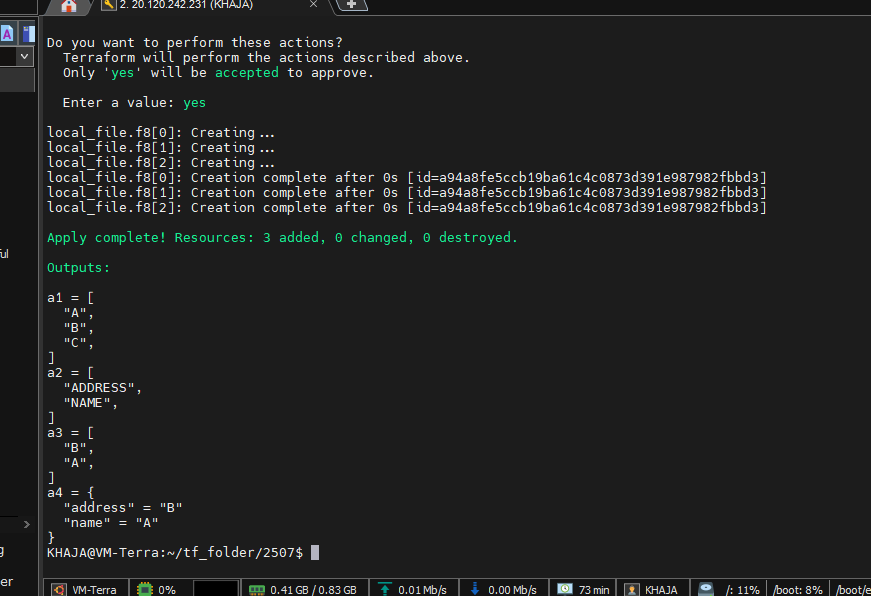


Run the command **terraform apply** to make the changes









**Terraform console**

The terraform console command provides an interactive shell for evaluating Terraform expressions and testing configurations. It's extremely useful for debugging and experimenting with Terraform code.

**1. Launching the Console**

terraform console

* This opens an interactive prompt where you can evaluate expressions.

**2. Basic Usage Examples**

**A. Evaluating Simple Expressions**

terraform

> 1 + 2

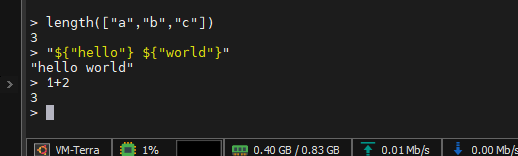
3

> "${"hello"} ${"world"}"

"hello world"

> length(["a", "b", "c"])

3



**Terraform functions**

Terraform provides built-in functions for transforming and combining values. Here's a comprehensive guide to all function categories with practical examples:

* additional capabilities to run/ execute in a smoother way

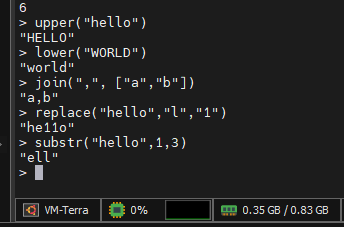
**1. Numeric Functions**

| **Function** | **Definition** | **Example** |
| --- | --- | --- |
| abs(x) | Absolute value | abs(-5) → 5 |
| ceil(x) | Rounds up to nearest integer | ceil(3.2) → 4 |
| floor(x) | Rounds down to nearest integer | floor(3.8) → 3 |
| max(N1,N2,...) | Returns highest value | max(5,9,2) → 9 |
| min(N1,N2,...) | Returns lowest value | min(5,9,2) → 2 |
| pow(x,y) | Raises x to power y | pow(2,3) → 8 |
| sum(list) | Sums all elements | sum([1,2,3]) → 6 |



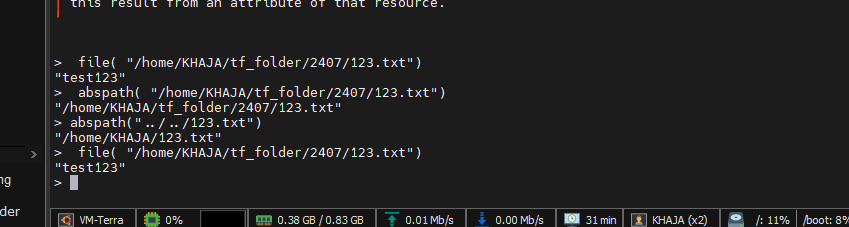
**2. String Functions**

| **Function** | **Definition** | **Example** |
| --- | --- | --- |
| upper(str) | Converts to uppercase | upper("hello") → "HELLO" |
| lower(str) | Converts to lowercase | lower("WORLD") → "world" |
| join(delim, list) | Combines list with delimiter | join(",", ["a","b"]) → "a,b" |
| replace(str,sub,new) | Replaces substring | replace("hello","l","1") → "he11o" |
| substr(str,start,len) | Extracts substring | substr("hello",1,3) → "ell" |



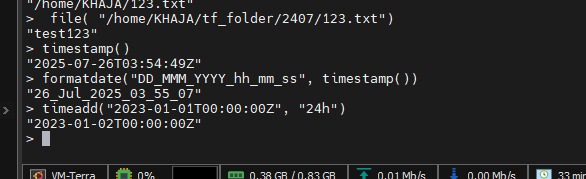
**3. Filesystem Functions**

| **Function** | **Definition** | **Example** |
| --- | --- | --- |
| abspath(path) | Converts to absolute path | abspath("./file") → "/home/project/file" |
| file(path) | Reads file contents | file("../../nginx.txt"→ file contents |
| fileexists(path) | Checks if file exists | fileexists("main.tf") → true |



**4. Date/Time Functions**

| **Function** | **Definition** | **Example** |
| --- | --- | --- |
| timestamp() | Current UTC timestamp | timestamp() → "2023-05-15T12:34:56Z" |
| formatdate(fmt,time) | Formats timestamp | formatdate("DD-MM-YYYY", timestamp()) → "15-05-2023" |
| timeadd(time,duration) | Adds time duration | timeadd("2023-01-01T00:00:00Z", "24h") → adds 1 day |



**5. Collection Functions**

| **Function** | **Definition** | **Example** |
| --- | --- | --- |
| length(collection) | Count of elements | length(["a","b"]) → 2 |
| toset(list) | Converts to unique set | toset(["a","b","a"]) → ["a","b"] |
| sort(list) | Alphabetical sort | sort(["b","a"]) → ["a","b"] |
| lookup(map,key,default) | Safe map lookup | lookup({a=1},"a",0) → 1 |
| contains(list,value) | Membership check | contains(["a","b"],"a") → true |
| distinct(list) | Removes duplicates | distinct([1,2,2]) → [1,2] |
| concat(lists...) | Combines lists | concat([1],[2]) → [1,2] |
| sum(list) | Numeric sum | sum([1,2,3]) → 6 |

