Lab 3. Terraform lists

Background:

Here, we learn how to create and use terraform lists data structures. We will learn now to create then, and how to extract individual elements of the lists and using looping constructs over the list.

Tasks:

1. Make a directory called ‘lab3’ underneath the terraform-labs directory.

2. Change into the directory.

3. Create the following files: *main.tf, resource.tf, variables.tf*

Here is the source code for the main.tf file:

```
provider "aws" {
    access_key = "AKIAIZAHH7GJN6ASXVVA"
    secret_key = "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"
    region = "${var.region}"

}
```

Here is the source code for the resource.tf file.

```
resource "aws_vpc" "main_vpc" {
    cidr_block = "${var.vpc_cidr}"
    instance_tenancy = "default"
    tags {
        Name = "Main"
        Location = "London"
    }
}

resource "aws_subnet" "vpc_subnets" {
    count = "${length(var.aws_availability_zones)}"
    vpc_id = "${aws_vpc.main_vpc.id}"
    cidr_block = "${element(var.vpc_subnet_cidr,count.index)}"
    availability_zone = "us-east-1a"
    tags {
        Name = "subnet-${count.index+1}"
   }
}

variable "region" {
    default =  "us-east-1"
}
```

Here is the source code for vars.tf:

```
variable "region" {
    default = "us-east-1"
}

variable "vpc_cidr" {
    default = "192.168.0.0/16"
}

variable "vpc_subnet_cidr" {
    type = "list"
    default =
["192.168.100.0/24","192.168.101.0/24","192.168.102.0/24"]
}

variable "ami_instance" {
    default = "ami-0ac019f4fcb7cb7e6"
}

variable "ami_instance_type" {
    default = "t2.micro"
}

variable "aws_availability_zones" {
    type = "list"
    default = ["us-east-1a","us-east-1b","us-east-1c"]
}
```

Note: For each student, change the CIDR for the VPC and subnets to a unique value!

4. Run the following commands:

```
> terraform get

 > terraform init
```

Note that the ‘>’ refers to the bash shell prompt and is not part of the command.

This command initializes the terraform directory structure.

7. Run the following command:

```
> terraform plan
```

This should print out what actions terraform will take.

8. Run the following command:

```
> terraform apply
```

Assuming that this works correctly, AWS create a VPC and three subnets. Check this with the AWS console.

9.. Run the following:

```
> terraform destroy
```

This will now destroy the formerly created AWS vpc, and all subnets.