**Terraform task  
Name : Varsha Yadav  
Batch 11  
Topic – Continuation Datatype Questions based on class   
  
Tuple Variables**

**Q1: What is a tuple type in Terraform, and how is it defined?**  
A1: A tuple type defines a fixed-length, ordered collection where each element has a specified type. For example:

variable "filename9" {

type = tuple([string, number, bool, list(number)])

default = ["adi", 123, true, [1,2,3]]

}

Each position must match the defined type exactly

**Q2: What are error scenarios for tuple values?**  
A2: Two common errors:

* Providing fewer or more elements than defined (e.g. ["adi", 123]).
* Mismatched types or wrong order, e.g. ["adi", 123, true, ["one",2,3]] where "one" isn't a number.

**Q3: How do you access nested tuple elements?**  
A3: Use index syntax:

var.filename9[3][1]

This accesses the 4th element (a list), then its 2nd element.

**Object Variables**

**Q4: What is an object type in Terraform?**  
A4: An object is a user-defined structured type where each attribute has its own name and type.  
Example:

variable "filename10" {

type = object({

name = string

id = number

address = list(string)

})

default = {

name = "adi"

id = 123

address = ["marathalli", "bangalore", "560037"]

}

}

``` :contentReference[oaicite:10]{index=10}

---

\*\*Q5: What are common object errors when injecting via `.tfvars` files?\*\*

A5: Errors occur when:

- A field has incorrect type: e.g. `"id" = "four"` instead of a number.

- Missing or extra keys: e.g. `{ house = "test", id = 456, address = [...] }` lacks `name` attribute.

---

\*\*Q6: How do you access elements in an object and its nested list?\*\*

A6:

- Access attributes via dot-syntax: `var.filename10.id`

- Nested list: `var.filename10.address[1]` returns `"bangalore"`.

Q&A with Resource Examples

\*\*Q7: Given this tuple, how do you reference the second number in the inner list of `filename9`?\*\*

A7: Use `var.filename9[3][1]`. With default `["adi",123,true,[1,2,3]]`, this returns `2`.

---

\*\*Q8:\*\* For `filename10`, how do you get the `id`, and how do you get the second address?

\*\*A8:\*\*

- `var.filename10.id` → returns `123`.

- `var.filename10.address[1]` → returns `"bangalore"`.

---

## tfvars Injection Errors

\*\*Q9: What errors occur when you provide malformed tuple values in `.tfvars` for `filename9`?\*\*

A9:

- `["adi",123"]` – missing the boolean and list elements → wrong count of elements.

- `["adi",123,true,["one",2,3]]` – type mismatch: `"one"` is string instead of number.

---

\*\*Q10: What happens if you supply a proper list of numbers but maintain types?\*\*

A10: `["adi",123,true,[4,5,6]]` is valid; element access like `var.filename9[3][2]` yields `6`.

---

## 🧰 Terraform Workspace Questions

\*\*Q11: What is a Terraform CLI workspace and what is the default workspace called?\*\*

A11: Terraform workspaces let you maintain multiple distinct states within the same configuration directory. The initial one is always named `default` :contentReference[oaicite:11]{index=11}.

---

\*\*Q12: How do you manage Terraform CLI workspaces? Provide commands.\*\*

A12: Use commands:

```bash

terraform workspace list

terraform workspace new <name>

terraform workspace select <name>

terraform workspace delete <name>

This allows you to switch between multiple state environments within the same configuration

**Q13: When should you *not* use CLI workspaces, and what are better alternatives?**  
A13: CLI workspaces are not ideal when you need isolated credentials or access controls per environment because they share the same backend. Better solutions: separate configurations each with its own backend, modules, or use terraform\_remote\_state for sharing data across configurations

**Q14: How can workspace names be used within Terraform configurations?**  
A14: Use terraform.workspace in expressions. Example:

hcl

CopyEdit

count = terraform.workspace == "default" ? 3 : 1

You can also interpolate workspace name into resource tags to distinguish deployments by workspace