**DAY 4** **[28 - JUNE - 2025]**

# TUPLE IN PYTHON

Tuples are used to store multiple items in a single variable. A tuple is a collection which is ordered and unchangeable. Tuples are written with round brackets.

*Tuple Items*

Tuple items are ordered, unchangeable, and allow duplicate values.

Tuple items are indexed, the first item has index [0], the second item has index [1] etc.

*Ordered*

When we say that tuples are ordered, it means that the items have a defined order, and that order will not change.

*Unchangeable*

Tuples are unchangeable, meaning that we cannot change, add or remove items after the tuple has been created.

*Allow Duplicates*

Since tuples are indexed, they can have items with the same value.

*Access Tuple Items*

You can access tuple items by referring to the index number, inside square brackets:

*Negative Indexing*

Negative indexing means start from the end.

-1 refers to the last item, -2 refers to the second last item etc.

*Range of Indexes*

You can specify a range of indexes by specifying where to start and where to end the range.

When specifying a range, the return value will be a new tuple with the specified items.

*Add Items*

Since tuples are immutable, they do not have a built-in append() method, but there are other ways to add items to a tuple.

*1. Convert into a list:* Just like the workaround for *changing* a tuple, you can convert it into a list, add your item(s), and convert it back into a tuple.

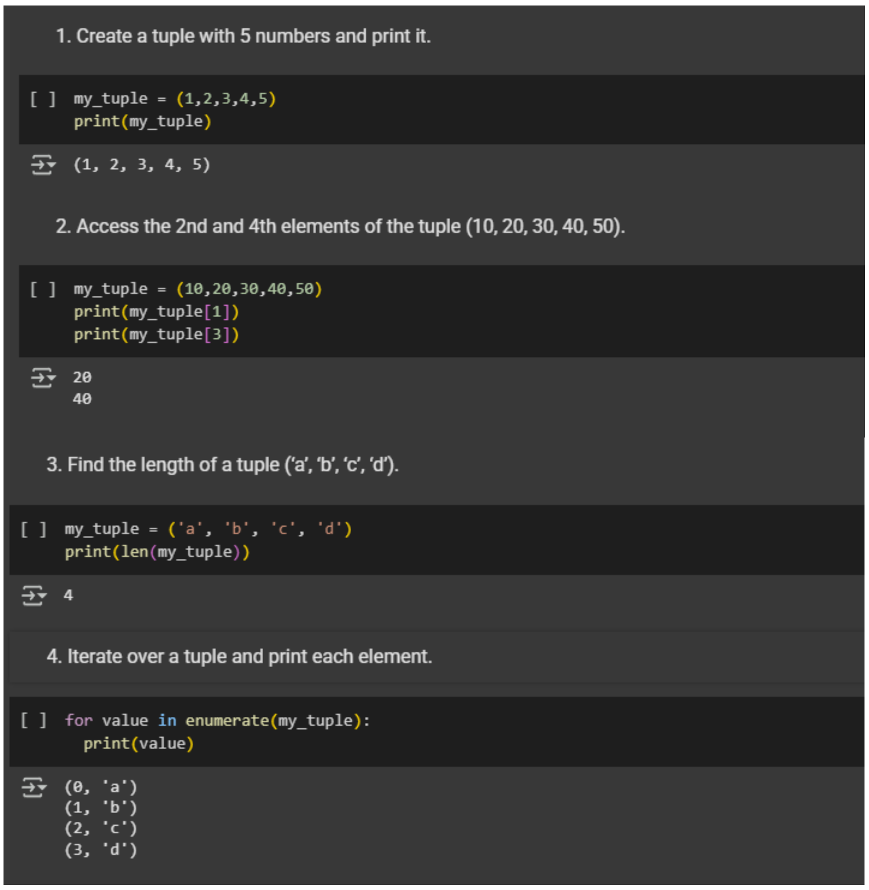
*2. Add tuple to a tuple.* You are allowed to add tuples to tuples, so if you want to add one item, (or many), create a new tuple with the item(s), and add it to the existing tuple: Using Asterisk*

If the number of variables is less than the number of values, you can add an * to the variable name and the values will be assigned to the variable as a list:

*Join Two Tuples*

To join two or more tuples you can use the + operator:

*PRACTICE QUESTIONS:*



1. Create a tuple with 5 numbers and print it.

```
[ ] my_tuple = (1,2,3,4,5)
    print(my_tuple)
```

```
(1, 2, 3, 4, 5)
```

2. Access the 2nd and 4th elements of the tuple (10, 20, 30, 40, 50).

```
[ ] my_tuple = (10,20,30,40,50)
    print(my_tuple[1])
    print(my_tuple[3])
```

```
20
40
```

3. Find the length of a tuple ('a', 'b', 'c', 'd').

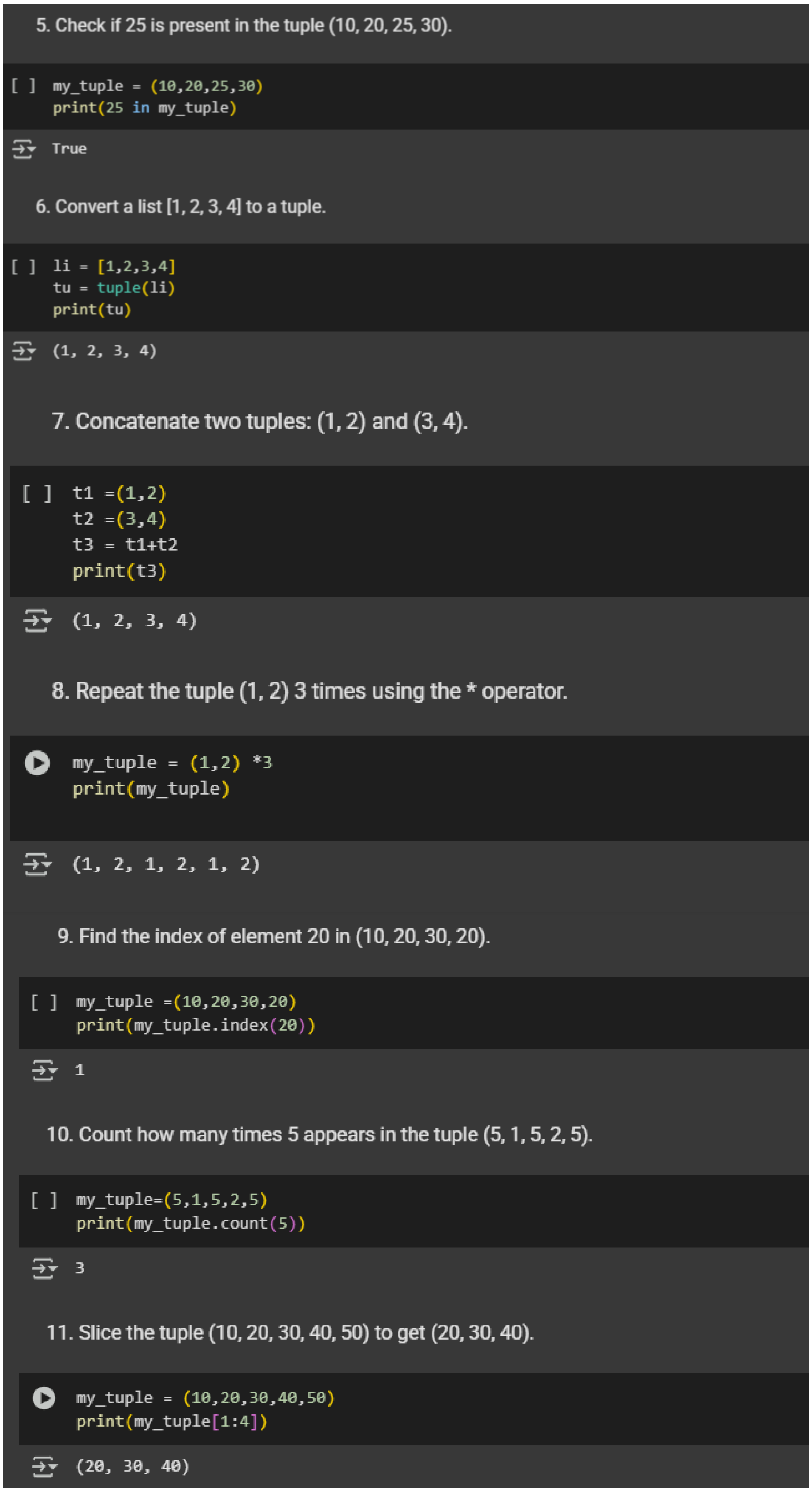
```
[ ] my_tuple = ('a', 'b', 'c', 'd')
    print(len(my_tuple))
```

```
4
```

4. Iterate over a tuple and print each element.

```
[ ] for value in enumerate(my_tuple):
      print(value)
```

```
(0, 'a')
(1, 'b')
(2, 'c')
(3, 'd')
```



5. Check if 25 is present in the tuple (10, 20, 25, 30).

```
my_tuple = (10,20,25,30)
print(25 in my_tuple)
```

```
True
```

6. Convert a list [1, 2, 3, 4] to a tuple.

```
li = [1,2,3,4]
tu = tuple(li)
print(tu)
```

```
(1, 2, 3, 4)
```

7. Concatenate two tuples: (1, 2) and (3, 4).

```
t1 =(1,2)
t2 =(3,4)
t3 = t1+t2
print(t3)
```

```
(1, 2, 3, 4)
```

8. Repeat the tuple (1, 2) 3 times using the * operator.

```
my_tuple = (1,2) *3
print(my_tuple)
```

```
(1, 2, 1, 2, 1, 2)
```

9. Find the index of element 20 in (10, 20, 30, 20).

```
my_tuple =(10,20,30,20)
print(my_tuple.index(20))
```

```
1
```

10. Count how many times 5 appears in the tuple (5, 1, 5, 2, 5).

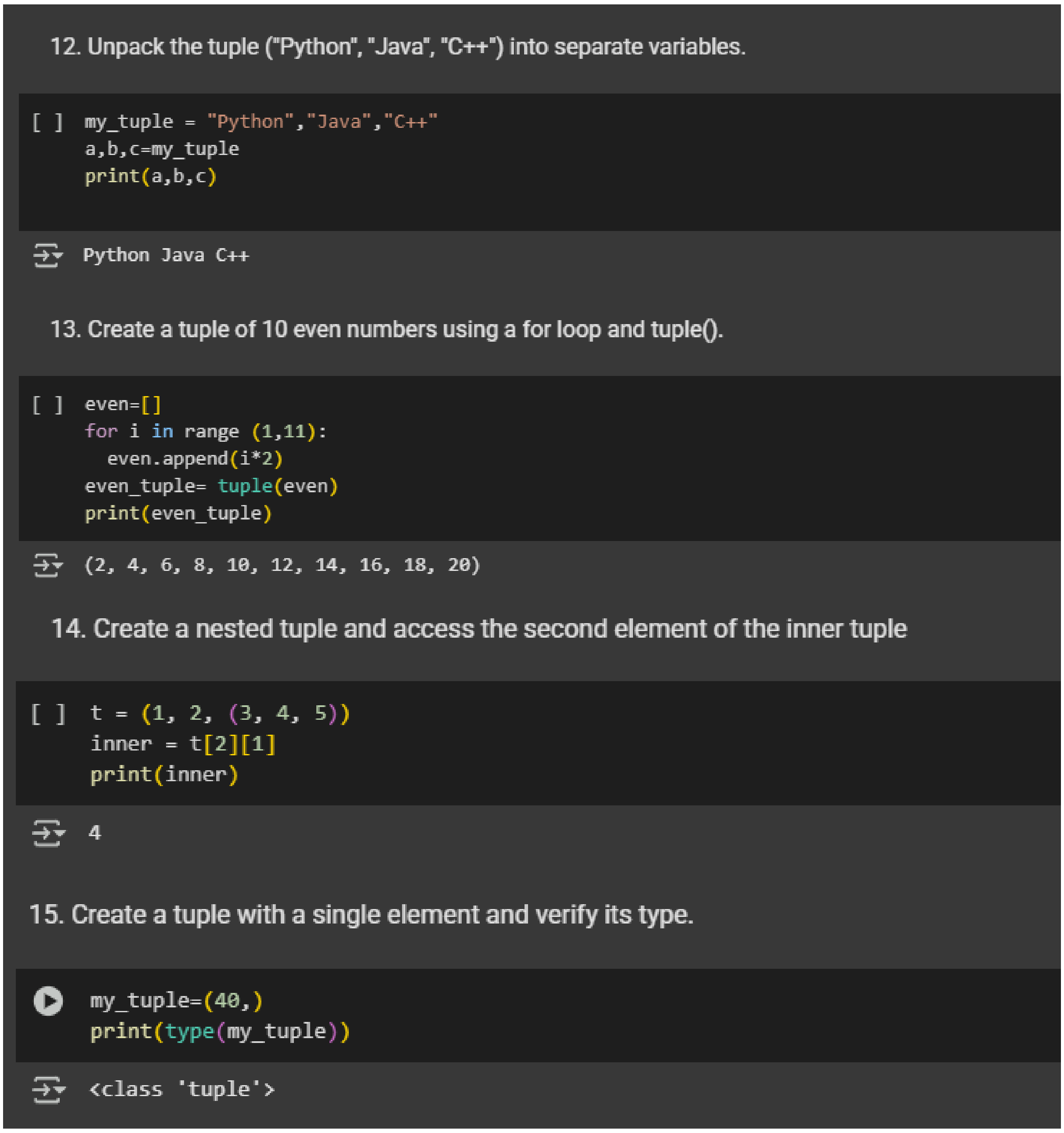
```
my_tuple=(5,1,5,2,5)
print(my_tuple.count(5))
```

```
3
```

11. Slice the tuple (10, 20, 30, 40, 50) to get (20, 30, 40).

```
my_tuple = (10,20,30,40,50)
print(my_tuple[1:4])
```

```
(20, 30, 40)
```



12. Unpack the tuple ("Python", "Java", "C++") into separate variables.

```
[ ] my_tuple = "Python","Java","C++"
    a,b,c=my_tuple
    print(a,b,c)
```

```
Python Java C++
```

13. Create a tuple of 10 even numbers using a for loop and tuple().

```
[ ] even=[]
    for i in range (1,11):
      even.append(i*2)
    even_tuple= tuple(even)
    print(even_tuple)
```

```
(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)
```

14. Create a nested tuple and access the second element of the inner tuple

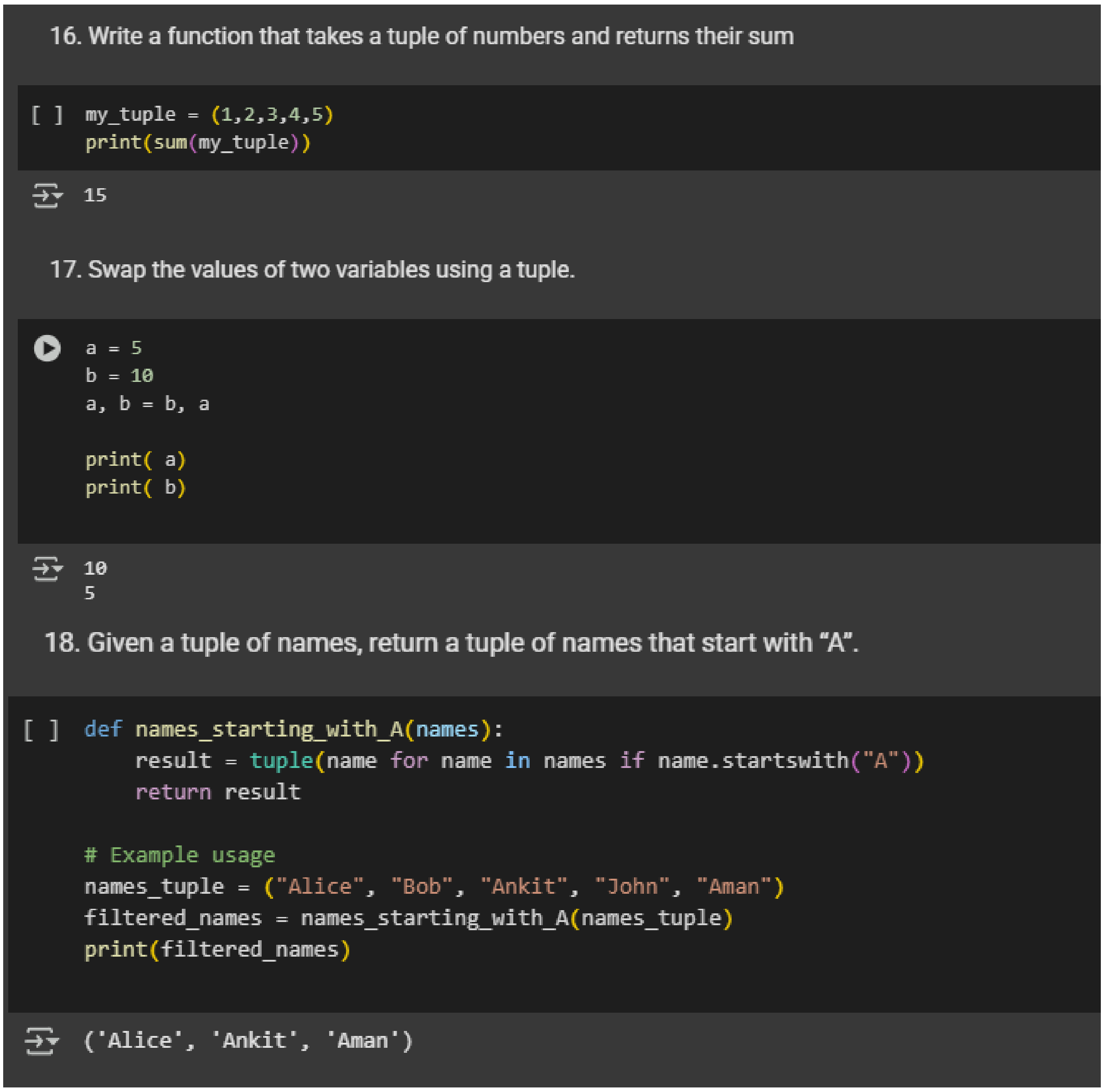
```
[ ] t = (1, 2, (3, 4, 5))
    inner = t[2][1]
    print(inner)
```

```
4
```

15. Create a tuple with a single element and verify its type.

```
my_tuple=(40,)
print(type(my_tuple))
```

```
<class 'tuple'>
```



16. Write a function that takes a tuple of numbers and returns their sum

```
my_tuple = (1,2,3,4,5)
print(sum(my_tuple))
```

```
15
```

17. Swap the values of two variables using a tuple.

```
a = 5
b = 10
a, b = b, a

print( a)
print( b)
```

```
10
5
```

18. Given a tuple of names, return a tuple of names that start with "A".

```
def names_starting_with_A(names):
    result = tuple(name for name in names if name.startswith("A"))
    return result

# Example usage
names_tuple = ("Alice", "Bob", "Ankit", "John", "Aman")
filtered_names = names_starting_with_A(names_tuple)
print(filtered_names)
```

```
('Alice', 'Ankit', 'Aman')
```