Python Tuple - Complete Explanation

A tuple in Python is an ordered, immutable collection of items. It is similar to a list but cannot be changed once created.

Basic Syntax:
my_tuple = (1, 2, 3, 'Python', 4.5)

Key Features:
• Ordered and indexed
• Immutable (cannot be changed)
• Allows duplicates
• Can hold different data types

Creation Methods:
1. Using (): my_tuple = (1, 2, 3)
2. Using tuple(): my_tuple = tuple([1, 2, 3])
3. Single-element tuple: my_tuple = (5,)
4. Empty tuple: my_tuple = ()

Accessing Elements:
print(my_tuple[0])
print(my_tuple[-1])

Slicing:
print(my_tuple[1:4])
print(my_tuple[::-1]) # Reverse

Immutability:
Once created, tuples cannot be modified. Example:
my_tuple[1] = 10 # ■ This will cause an error

Looping Through Tuple:
for item in my_tuple:
print(item)

Tuple Operations:
• Concatenation: (1, 2) + (3, 4)
• Repetition: (1, 2) * 3
• Membership: 2 in (1, 2, 3)
• Length: len(my_tuple)
• Counting and Indexing: my_tuple.count(2), my_tuple.index(3)

Tuple Packing and Unpacking:
person = ('Gaurav', 21, 'CSE')
name, age, branch = person

Nested Tuples:
nested = ((1, 2), (3, 4))
print(nested[1][0]) # Output: 3

Advantages of Tuples Over Lists:
• Faster performance (due to immutability)
• Can be used as keys in dictionaries (lists cannot)
• Protects data from accidental modification

Example - Returning Multiple Values from a Function:
def stats(a, b):
return (a + b, a * b)

```
result = stats(3, 4)
print(result) # Output: (7, 12)
```

Summary:
Create: t = (1, 2, 3)
Access: t[0]
Length: len(t)
Count: t.count(2)
Index: t.index(3)
Combine: t1 + t2
Repeat: t * 2
Unpack: a, b = (1, 2)