Tuples in Python

## Definition

A tuple is an immutable sequence type in Python that allows you to store a collection of ordered elements. Once a tuple is created, its elements cannot be modified (i.e., they cannot be added, removed, or changed).

## Characteristics of Tuples

- Ordered: Elements have a defined order, and you can access elements by index.  
- Immutable: Once created, elements cannot be changed, added, or removed.  
- Allow Duplicates: A tuple can have multiple occurrences of the same element.  
- Can store elements of different data types (int, float, string, etc.).

## Properties of Tuples

- Defined using parentheses () or just by separating items with commas.  
- The length of a tuple is fixed.  
- Tuples are hashable if all elements are hashable, which makes them usable as keys in dictionaries.

## Tuple Operations

## Creation

# Creating tuples  
tuple1 = (1, 2, 3)  
tuple2 = ("apple", "banana", "cherry")  
tuple3 = (1, "hello", 3.14)  
empty\_tuple = ()  
single\_element\_tuple = (5,) # Note the comma

## Accessing Elements

fruits = ("apple", "banana", "cherry")  
print(fruits[0]) # Output: apple  
print(fruits[-1]) # Output: cherry

## Slicing

numbers = (1, 2, 3, 4, 5)  
print(numbers[1:4]) # Output: (2, 3, 4)  
print(numbers[:3]) # Output: (1, 2, 3)

## Tuple Concatenation

tuple1 = (1, 2, 3)  
tuple2 = (4, 5)  
combined = tuple1 + tuple2  
print(combined) # Output: (1, 2, 3, 4, 5)

## Repetition

tuple1 = ("a", "b")  
repeated = tuple1 \* 3  
print(repeated) # Output: ('a', 'b', 'a', 'b', 'a', 'b')

## Membership Test

fruits = ("apple", "banana", "cherry")  
print("banana" in fruits) # Output: True  
print("grape" in fruits) # Output: False

## Tuple Unpacking

person = ("John", 25, "Engineer")  
name, age, profession = person  
print(name) # Output: John  
print(age) # Output: 25  
print(profession) # Output: Engineer

## Length of Tuple

numbers = (1, 2, 3, 4, 5)  
print(len(numbers)) # Output: 5

## Why Use Tuples?

- Faster than lists due to immutability.  
- Protect data from accidental modifications.  
- Can be used as dictionary keys if elements are hashable.

## Example Program: Basic Tuple Operations

# Example Program to demonstrate tuple operations  
  
# Creating tuples  
fruits = ("apple", "banana", "cherry", "apple")  
numbers = (1, 2, 3, 4, 5)  
  
# Accessing elements  
print("First fruit:", fruits[0])  
  
# Slicing  
print("Fruits from index 1 to 3:", fruits[1:3])  
  
# Concatenation  
combined = fruits + ("grape", "mango")  
print("Combined tuple:", combined)  
  
# Repetition  
print("Repeated tuple:", ("hello",) \* 2)  
  
# Membership test  
print("Is 'banana' in fruits?", "banana" in fruits)  
  
# Tuple unpacking  
name, age, profession = ("Alice", 30, "Developer")  
print("Name:", name)  
print("Age:", age)  
print("Profession:", profession)  
  
# Length of a tuple  
print("Number of numbers:", len(numbers))  
```