



# What is a Tuple?

A **tuple** is an **ordered, immutable** (unchangeable) collection of items. It is written with **parentheses ()**, e.g.

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

---



## 1. Creating Tuples

```
t1 = (1, 2, 3)
t2 = ("apple", "banana", "cherry")
t3 = (1, "hello", 3.5, True)
t4 = ()           # Empty tuple
t5 = (5,)         # Single element tuple (note the comma)
```

---



## 2. Accessing Elements (Indexing)

```
t = (10, 20, 30, 40, 50)
print(t[0])    # 10   (first element)
print(t[-1])   # 50   (last element)
```

---



## 3. Slicing Tuples

```
t = (1, 2, 3, 4, 5, 6)
print(t[1:4])   # (2, 3, 4)
print(t[:3])    # (1, 2, 3)
print(t[::2])   # (1, 3, 5)
```

---



## 4. Tuple Concatenation

```
t1 = (1, 2, 3)
t2 = (4, 5, 6)
t3 = t1 + t2
print(t3)      # (1, 2, 3, 4, 5, 6)
```

---



## 5. Repetition

```
t = (1, 2)
print(t * 3)    # (1, 2, 1, 2, 1, 2)
```

---

## 6. Membership Test

```
t = (10, 20, 30)
print(20 in t)      # True
print(40 not in t)  # True
```

---

## 7. Tuple Length

```
t = (5, 10, 15, 20)
print(len(t))      # 4
```

---

## 8. Min, Max, and Sum (for numeric tuples)

```
t = (3, 7, 2, 9)
print(min(t))      # 2
print(max(t))      # 9
print(sum(t))      # 21
```

---

## 9. Converting Between Tuples and Lists

```
t = (1, 2, 3)
lst = list(t)
lst.append(4)
t = tuple(lst)
print(t)           # (1, 2, 3, 4)
```

---

## 10. Tuple Packing and Unpacking

```
# Packing
t = (1, "apple", 3.5)

# Unpacking
(a, b, c) = t
print(a)    # 1
print(b)    # apple
print(c)    # 3.5
```

---

## 11. Built-in Tuple Methods

Tuples have only **two methods**:

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

print(t.count(2))    # 2 (counts how many times 2 appears)
print(t.index(3))    # 3 (returns index of first occurrence of 3)
```

---

## 12. Nested Tuples

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

---

## 13. Immutability Check

Tuples **cannot** be changed after creation.

```
t = (1, 2, 3)
# t[0] = 100    ✗ Error: TypeError (cannot modify a tuple)
```

---

## 14. Iterating Over a Tuple

```
t = ("red", "green", "blue")
for color in t:
    print(color)
```

---

## 15. Using Tuples in Functions

Tuples are often returned from functions:

```
def stats():
    return (10, 20, 30)

a, b, c = stats()
print(a, b, c)
```

---

## Summary Table

Operation	Example	Result
Indexing	<code>t[0]</code>	First element
Slicing	<code>t[1:3]</code>	Subset
Concatenation	<code>t1 + t2</code>	Combine tuples
Repetition	<code>t * 2</code>	Repeat items
Membership	<code>x in t</code>	True/False
Length	<code>len(t)</code>	Number of elements
Count	<code>t.count(x)</code>	Occurrences
Index	<code>t.index(x)</code>	First position
Min/Max/Sum	<code>min(t)</code>	Math ops

Operation	Example	Result
Conversion	<code>tuple(list)</code>	Change type
Unpacking	<code>a,b = t</code>	Assign values