Python Data Types (Primitive vs Non-Primitive)

1. Primitive Data Types

Primitive data types are the basic building blocks that cannot be broken down further.

■ *Integer (int)*

```
x = 10
y = -25
print(x, type(x))  # 10 <class 'int'>
```

■ *Float (float)*

```
price = 99.99
print(price, type(price))  # 99.99 <class 'float'>
```

■ *Complex (complex)*

```
z = 3 + 4j
print(z.real, z.imag)  # 3.0 4.0
```

■ *Boolean (bool)*

```
is_active = True
print(is_active, type(is_active))  # True <class 'bool'>
```

■ *String (str)*

```
name = 'Python'
print(name.upper())  # PYTHON
```

2. Non-Primitive (Composite) Data Types

Non-primitive data types are derived types that can hold multiple values or combine different primitives.

■ *List (list)*

```
fruits = ['apple', 'banana', 'cherry']
fruits.append('mango')
print(fruits)  # ['apple', 'banana', 'cherry', 'mango']
```

■ *Tuple (tuple)*

```
coordinates = (10.5, 20.3)
print(coordinates[0])  # 10.5
```

■ *Dictionary (dict)*

```
student = {'name': 'Alice', 'age': 22}
print(student['name'])  # Alice
```

■ *Set (set)*

```
nums = {1, 2, 2, 3, 4}
print(nums)  # {1, 2, 3, 4}
```

■ *Frozenset (frozenset)*

```
fset = frozenset([1, 2, 3, 2])
print(fset)  # frozenset({1, 2, 3})
```

3. Type Conversion (Casting)

Python allows conversion between primitive and non-primitive types. Example:

```
x = '100'
print(int(x))   # 100
print(float(x)) # 100.0

nums = [1, 2, 2, 3]
print(set(nums))  # {1, 2, 3}
```

4. Real-Life Examples

■ *Salary Calculation (Primitive)*

```
salary = 50000
bonus = 5000.50
total = salary + bonus
print('Total:', total)
```

■ *Shopping Cart (Non-Primitive)*

```
cart = {'apple': 3, 'banana': 5, 'milk': 2}
print('Items in cart:', cart)
```

■ *Unique Visitors (Set)*

```
visitors = [101, 102, 101, 103]
unique_visitors = set(visitors)
print('Unique Visitors:', unique_visitors)
```

■ Summary

Primitive Data Types: int, float, complex, bool, str
Non-Primitive Data Types: list, tuple, dict, set, frozenset