### 1. Introduction to Python Programming and Data Types

#### \*\*Introduction to Python\*\*

- Python is a high-level, interpreted programming language.

- It is known for its simplicity, readability, and ease of use.

- Used in various fields such as web development, data science, machine learning, and automation.

- Python is platform-independent and has a large community with extensive libraries.

#### \*\*Data Types in Python\*\*

- \*\*Numeric Types\*\*: `int` (integer), `float` (decimal numbers), `complex` (complex numbers).

- Example:

```python

x = 10 # int

y = 3.14 # float

z = 2 + 3j # complex

```

- \*\*Text Type\*\*: `str` (string). Used for textual data.

- Example:

```python

name = "Alice"

```

- \*\*Boolean Type\*\*: `bool` (True or False).

- Example:

```python

is\_active = True

```

- \*\*Sequence Types\*\*: `list`, `tuple`, `range`.

- Example:

```python

my\_list = [1, 2, 3]

my\_tuple = (4, 5, 6)

my\_range = range(10)

```

- \*\*Mapping Type\*\*: `dict` (dictionary).

- Example:

```python

my\_dict = {"key": "value"}

```

- \*\*Set Types\*\*: `set`, `frozenset`.

- Example:

```python

my\_set = {1, 2, 3}

```

- \*\*None Type\*\*: `NoneType` (represents no value).

- Example:

```python

value = None

```

---

### 2. Operators

#### \*\*Types of Operators\*\*

1. \*\*Arithmetic Operators\*\*: Perform mathematical operations.

- Examples: `+`, `-`, `\*`, `/`, `//`, `%`, `\*\*`.

```python

a = 10

b = 3

print(a + b) # 13

print(a \*\* b) # 1000 (10 to the power 3)

```

2. \*\*Comparison Operators\*\*: Compare values and return a boolean.

- Examples: `<`, `>`, `<=`, `>=`, `==`, `!=`.

```python

print(a > b) # True

print(a == b) # False

```

3. \*\*Logical Operators\*\*: Combine multiple conditions.

- Examples: `and`, `or`, `not`.

```python

x = True

y = False

print(x and y) # False

print(x or y) # True

```

4. \*\*Assignment Operators\*\*: Assign and modify values.

- Examples: `=`, `+=`, `-=`, `\*=`, `/=`, `%=`, `\*\*=`.

```python

total = 10

total += 5

print(total) # 15

```

5. \*\*Membership Operators\*\*: Check membership in sequences.

- Examples: `in`, `not in`.

```python

my\_list = [1, 2, 3]

print(2 in my\_list) # True

```

6. \*\*Identity Operators\*\*: Check if two objects are the same.

- Examples: `is`, `is not`.

```python

a = [1, 2]

b = a

print(a is b) # True

```

---

### 3. Conditional Handling and Control Statements

#### \*\*Conditional Statements\*\*

1. \*\*if Statement\*\*: Executes a block if the condition is True.

```python

x = 10

if x > 5:

print("x is greater than 5")

```

2. \*\*if-else Statement\*\*: Executes one block if the condition is True, otherwise another block.

```python

if x % 2 == 0:

print("x is even")

else:

print("x is odd")

```

3. \*\*if-elif-else Statement\*\*: Multiple conditions.

```python

if x < 5:

print("x is less than 5")

elif x == 5:

print("x is 5")

else:

print("x is greater than 5")

```

---

### 4. Loops

#### \*\*Types of Loops\*\*

1. \*\*for Loop\*\*: Iterates over a sequence.

```python

for i in range(5):

print(i)

```

2. \*\*while Loop\*\*: Executes as long as the condition is True.

```python

count = 0

while count < 5:

print(count)

count += 1

```

#### \*\*Loop Control Statements\*\*

- \*\*break\*\*: Exits the loop.

```python

for i in range(5):

if i == 3:

break

print(i)

```

- \*\*continue\*\*: Skips the current iteration.

```python

for i in range(5):

if i == 2:

continue

print(i)

```

---

### 5. Keywords

#### \*\*What are Keywords?\*\*

- Keywords are reserved words in Python with specific meanings.

- They cannot be used as variable names.

#### \*\*Examples of Keywords\*\*

- \*\*Control flow\*\*: `if`, `else`, `elif`, `while`, `for`, `break`, `continue`.

- \*\*Boolean\*\*: `True`, `False`.

- \*\*Variable handling\*\*: `None`, `global`, `nonlocal`, `del`.

- \*\*Class and Function\*\*: `class`, `def`, `return`, `yield`.

- \*\*Importing\*\*: `import`, `from`, `as`.

#### Example:

```python

if True:

print("This is a keyword example")

```