Introduction and Conditional Statements

1. Programming Cycle for Python

The programming cycle in Python involves the following steps:

- **Problem Analysis:** Understand the problem and define the requirements.
- **Algorithm Design:** Create a step-by-step plan to solve the problem.
- **Coding:** Write the program in Python using an IDE or text editor.
- **Testing:** Run the program to check for errors and verify the output.
- **Debugging:** Identify and fix errors in the code.
- **Documentation:** Add comments and explanations for future reference.
- **Maintenance:** Update and improve the program as needed.

2. Python IDE

IDE (Integrated Development Environment): *A software application that provides tools for coding, debugging, and testing.*

Popular Python IDEs:

- **IDLE:** Default Python IDE.
- **PyCharm:** Advanced IDE with debugging and project management.
- **VS Code:** Lightweight and customizable editor.
- **Jupyter Notebook:** Interactive environment for data science.

3. Interacting with Python Programs

Interactive Mode: *Run Python commands directly in the Python shell.*

- Example: Type `python` in the terminal to start the shell.

Script Mode: *Write code in a `.py` file and execute it.*

- Example: Save code in `script.py` and run it using `python script.py`.

4. Elements of Python

Variables: *Used to store data.*

- Example: `x = 10`.

Data Types: *Integers (`int`), Floats (`float`), Strings (`str`), Booleans (`bool`), etc.*

Comments: *Use `#` for single-line comments and `"` or `"""` for multi-line comments.*

Indentation: *Python uses indentation(spaces) to define code blocks.*

5. Type Conversion

Implicit Conversion: *Automatically performed by Python.*

- Example: `x = 10 + 5.5` (int + float → float).

Explicit Conversion: *Done using built-in functions.*

- Example: `x = int("10")`.

`int()`: *Converts to integer.*

`float()`: *Converts to float.*

`str()`: *Converts to string.*

Basics

1. Expression

A combination of values, variables, and operators that evaluates to a single value.

- Example: `x = 5 + 3 * 2` (result: 11).

2. Assignment Statement

Used to assign a value to a variable.

- Example: `x = 10`.

Multiple assignments:

- Example: `x, y = 5, 10`.

3. Arithmetic Operators

`+` (Addition), `-` (Subtraction), `*` (Multiplication), `/` (Division), `%` (Modulus), `**` (Exponentiation), `//` (Floor Division).

- Example: `x = 10 // 3` (result: 3).

4. Operator Precedence

Determines the order of operations in an expression.

Order:

I. Parentheses
II. Exponentiation
III. Multiplication/Division
IV. Addition/Subtraction.

Written By **Krishna Singh**

- Example: `x = 5 + 3 * 2` (result: 11, not 16).

5. Boolean Expression

An expression that evaluates to `True` or `False`.

Relational Operators: *`==`, `!=`, `>`, `<`, `>=`, `<=`.*

- Example: `x = 5 > 3`
  (result: `True`).

Logical Operators: *`and`, `or`, `not`.*

- Example: `x = (5 > 3) and (2 < 4)`
  (result: `True`).

Conditionals

1. Conditional Statements in Python

if Statement: *Executes a block of code if the condition is true.*

```
if condition:
    # Code to execute
```

if-else Statement: *Executes one block if the condition is true, else another block.*

```
if condition:
    # Code to execute if true
else:
    # Code to execute if false
```

2. Nested-if Statement

An `if` statement inside another `if` statement.

```
if condition1:
    if condition2:
        # Code to execute
```

Written By **Krishna Singh**

3. ‘elif’ Statement

Used to check multiple conditions.

```
if condition1:
    # Code to execute
elif condition2:
    # Code to execute
else:
    # Code to execute
```

4. Expression Evaluation

Python evaluates expressions based on operator precedence and associativity.

- Example: `x = 5 + 3 * 2` (result: 11).

5. Float Representation

Floating-point numbers are represented with a decimal point. Precision issues can occur due to binary representation.

- Example: `0.1 + 0.2` may not equal `0.3` exactly.