Unit 1: Introduction to Python Programming

Topic: Introduction to Python, Variables in Python, Data Types in Python, Conditional Statements, Loops in Python, Lists, Tuples, Sets, and Dictionaries.

1.Introduction to Python

1.1 What is Python?

Python is a high-level, interpreted, and general-purpose programming language created by Guido van Rossum in 1991. It is known for its simple and readable syntax, making it easy for beginners to learn and use. Python is widely used to develop websites, applications, software, and for data analysis, and is suitable for both students and professionals.

1.2 Features of Python

- Easy to learn and use
- Open source and free
- Platform independent
- Object-Oriented language
- Large standard library
- Used in AI, Data Science, Web Development, Automation, etc.

1.3 Applications of Python

- Web Development (Django, Flask)
- Data Science & Machine Learning
- Instagram uses Python
- YouTube uses Python
- Software Development
- Automation/Scripting
- Game Development
- Networking

1.4 First Python Program

```
print("Hello World")
```

2. What is Operator?

Ans. Operators are symbols that tell Python to perform actions like addition, comparison, or logical checking.

- **Types of Operators**
  a. **Arithmetic Operators:**
     Used for mathematical calculations. (+, -, *, /, %, **)

  b. **Relational (Comparison) Operators:**
     Used to compare two values. (==, !=, >,<, >=, <=)

  c. **Logical Operators:**
     Used to combine conditions.(and , or, not)

  d. **Assignment Operators:**
     Used to assign values.(=, +=, -=, *=, /=, %=)

2.1 Input in Python

input() : input() statement is used to accept values (using keyboard) from user.

- input() : result for input() is always a str.
- int(input()): int
- float(input()): float

3. What is Keywords?

Ans. Python has a set of keywords that are reserved words that cannot be used as variable names, function names, or any other identifiers:

Below are the Keywords:
False, None, True, and, as, assert, break, class, continue, def, del, elif, else, except, finally, for, from, global, if, import, in, is, lambda, nonlocal, not, or, pass, raise, return, try, while, with, yield

4. What is Comments ?

Ans. Comments are notes written inside a program to explain what the code is doing.

4.1 Types of Comments in Python:

- **Single line comments:** Starts with #
- **Multi-Line Comment:** Python does not have a special symbol, but we use triple quotes.

5. Basic Programming Concepts

Every program follows three basic steps:

- **Input**
  - Taking data from the user.
    name = input("Enter your name: ")

- **Process**
  - Performing operations on the input.
    marks = int(input("Enter marks: "))
    result = marks + 5

- **Output**
  - Displaying the result.
    print(result)

6. Variables in Python

Definition

A variable is a name given to a memory location used to store data.

- **Rules for Naming Variables**
  - Must start with a letter or underscore
  - Cannot start with a number
  - Cannot use keywords
  - Case-sensitive

Examples:

```
a = 10
name = "Rahul"
price = 99.5
```

- Python automatically decides the type of variable.

7. Data Types in Python

Python supports different types of data:

7.1 Integer (int) : Stores whole numbers.

Example:

```
x = 10
y = 20
```

7.2 Float (float) : Stores decimal numbers.

Example:

```
y = 3.14
x = 4.53
```

7.3. Boolean(bool): Stores True or False.

Example:

```
is_pass = True
```

7.8 Type Checking: Used to check which type of data is stored in a given variable.

Example:

```
x = 10
print(type(x))
```

8. What is a String?

Ans. A string in Python is a sequence of characters enclosed in single quotes, double quotes. It is used to store and manipulate text data.

Example:

```
name = “Shiv"
message = 'Hello World’
```

- Here, “Shiv" and 'Hello World' are strings.

8.1 How to Create a String?

Ans. You can create a string using:

```
Single quotes -- 'Hello’
Double quotes -- "Hello"
```

Example:

```
text1 = 'Python’
text2 = "Programming"
text3 = “This is a multi-line string”
```

8.2 Accessing Characters (Indexing):- Each character in a string has a position (index).

Example:

```
word = "Python"
print(word[0])   # P
print(word[3])   # h
```

Index starts from 0

8.1 String Length:- Use len() to find the number of characters.

Example:

```
name = "Prince"
print(len(name))  # 6
```

8.2 String Concatenation (Joining Strings)

Example:

```
first = “Shiv"
last = "Kumar"
print(first + " " + last)
```

Output: Shiv Kumar

8.3 Strings are Immutable:

Once created, a string cannot be changed directly.

Example:

```
name = "Prince"
name[0] = "p"  Error
```

- You must create a new string instead.

8.4 Important String Methods

- **upper() –** Convert to capital letters
  Example:
  ```
  text = "hello"
  print(text.upper())
  ```
  Output: HELLO

- **lower()** – Convert to small letters
  Example:
  ```
  text = "HELLO"
  print(text.lower())
  ```
  Output: hello

- **len() –** Find length of string
  Example:
  ```
  text = "Python"
  print(len(text))
  ```
  Output: 6

- **count()** – Count a character
  Example:
  ```
  text = "banana"
  print(text.count('a'))
  ```
  Output: 3

- **find() –** Find position of character/word
  Example:
  ```
  text = "Hello World"
  print(text.find("World"))
  ```
  Output: 6

- **replace()** – Replace word
  Example:

  ```
  text = "I like Java"
  print(text.replace("Java", "Python"))
  ```

  Output: I like Python

- **strip() –** Remove extra spaces
  Example:

  ```
  text = "  hello  "
  print(text.strip())
  ```

  Output: hello

- **split() –** Convert string to list
  Example:

  ```
  text = "apple,banana,mango"
  print(text.split(","))
  ```

  Output: ['apple', 'banana', 'mango']

9. Conditional Statements: Used for decision making.

9.1 if Statement

Example:

```
age = 20
if age >= 18:
print("Eligible to vote")
```

9.2 if-else Statement

Example:

```
marks = 35
if marks >= 40:
   print("Pass")
else:
   print("Fail")
```

9.3 if-elif-else Statement

Example:

```
marks = 75

if marks >= 90:
  print("Grade A")
elif marks >= 60:
  print("Grade B")
else:
  print("Grade C")
```

10. Loops in Python: Loops are used to repeat tasks.

10.1 for Loop: Used when number of repetitions is known.

Example:

```
for i in range(1,6):
      print(i)
```

10.2 while Loop: Used when repetitions depend on condition.

Example:

```
i = 1
while i <= 5:
  print(i)
  i += 1
```

11 . Lists

Definition: A list is an ordered collection of items.

Features

- Changeable (mutable)
- Allows duplicate values
- Uses square brackets []

Example

```
numbers = [10, 20, 30, 40]
print(numbers[0])
```

Operations on List

```
numbers.append(50)
numbers.remove(20)
```

12. Tuples

Definition: A tuple is an ordered collection that cannot be changed (immutable).

Features

- Fixed size
- Faster than list
- Uses ()

Example

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

13. Sets

Definition: A set is an unordered collection of unique elements.

Features

- No duplicates allowed
- No indexing
- Uses {}

Example

```
s = {1, 2, 3, 3, 4}
print(s)
Output: {1, 2, 3, 4}
```

14. Dictionaries

Definition: A dictionary stores data in key-value pairs.

Features

- Mutable
- Fast access using keys
- Uses {}

Example

```
student = {
"name": "Aman",
"age": 21,
"branch": "CSE"
}

print(student["name"])
```

15. Difference Between List, Tuple, Set, Dictionary

Feature	**List**	**Tuple**	**Set**	**Dictionary**
Ordered	Yes	Yes	No	Yes
Mutable	Yes	No	Yes	Yes
Duplicates	Yes	Yes	No	Keys not duplicate
Syntax	[]	()	{}	{key:value}

Programming Questions

- Program to check even/odd.
- Program to find sum using loop.
- Program using list operations.
- Program to store student data in dictionary.

****