Date : 16/08/25

MON TUE WED THU FRI SAT SUN

□□□□□□□

Module - 1 : Assignment Questions.

Python Basics : Assignment Questions - 1

① what is the python interpreter and what is the difference between interpreter and compiler ?

A→ Python Interpreter : A program that executes python code line - by - line.

Interpreter V/S Compiler :

- Interpreter executes code line-by-line → Slower but eazy to debug.
- Compiler translates the whole code into machine code at once → faster execution.
- Python uses interpreter, not compiler.

② Explain the Difference between the python interactive mode and script mode.

A→ • Interactive mode :
- Code is typed directly in python shell. (>>>)
- Executes immediately, good for small tests.

- Script mode :
  - Code is written in (.py) file and executed later.
  - Suitable for larger programs.

(3) What is console input/output in python? Give examples of input and output functions.

- Console input : Taking user input using "input ()".

eg :
name = input ("Enter your name:")

- Console output: Displaying output using "print ()".

eg:
print ("Hello", name")

(4) what is the purpose of the "input()" function in python?

★ • Used to take input from user as a string.

- Example : age = input ("enter your age:")

⑤ Define conditions in python. How is an if statement used?

A->• Conditions : logical expressions that evaluate to true/false.

• if statement example :

```
x = 10
if x > 5 :
    print ("x is greater than 5")
```

⑥ Differentiate between if, if-else, and if - elif - else statements with examples.

• if : Runs if condition is true.

eg:
```
if x > 0 : print (" Positive")
```

• if -else : Two - way decision

eg:
```
if x > 0 : print ("positive")
else : print (" Negative or Zero ")
```

• if - elif - else : Multiple Conditions.

```
if x > 0 : print (" positive")
elif x == 0 : print ("zero")
else : print ("Negative")
```

7. What is control flow in python? Give examples of control flow statemen

Ans• Control flow: Determines the order of execution of codes.

- Examples:
  - Conditional → if, if-else, if-elif-else.
  - loops → for, while.
  - jump → Break, continue, pass.

8. What is the difference between for loop and while loop in python?

- for loop: Iterates over sequence (fixed repetitions)

eg.
```
for i in range(5): print (i)
```

- While loop: Runs until condition is false.

eg.
```
i=0
while i<5:
    print (i)
    i+=1
```

9. Define a function in python. How do you define and call a function?

Ans• Function: A reusable block of code.

• Define :

```
def greet():
    print("Hello")
```

• Call :

```
greet()
```

(10) What is the difference between built-in functions and user-defined functions?

Ans • Built-in Functions : Predefined (eg., print(), len(), type() ..).

• User defined functions: Written by programmers using "def".

(11) What are python datatypes? Name five builtin datatypes with examples.

• Datatypes : Define type of data stored.

• Examples:

1. int → (x = 10)
2. float → (y = 3.14)
3. str → (name = "pranay")
4. bool → (flag = True)
5. list → (nums = [1, 2, 3])

(12) What is a list in python? Write its main features.

Ans • list : Ordered, mutable collection.

- Features :
  - written in [].
  - Can store mixed datatypes.
  - Allows indexing / slicing.
  - Example : fruits = ["apple", "banana", "mango"]

(13) Define a tuple in python. How is it different from a list?

Ans • Tuple : ordered, immutable collection.

- Syntax : t = (1, 2, 3)

- Difference :
  - list → mutable (can be changed,
  - Tuple → immutable (cannot be changed

(14) What is a set in python? How is it different from a list or tuple?

Ans • Set : Unordered collection of unique items.

- Syntax : s = {1, 2, 3}

- Differences:
  - No Duplicates
  - Unordered (no indexing)
  - Faster membership testing than list / tuple.

(15) What is file I/O in python? Explain the use of open() function with modes.

- File I/O : Reading and writing files.

- open() function : Opens a file in specific mode.
  - "r" → Read (Default)
  - "w" → Write (overwrites)
  - "a" → Append.
  - ("rb"/"wb") → Binary modes

- Example:

```
f = open ("data.txt, "r")
content = f.read ()
f.close ().
```