**Module 15 : Advance Python Programming**

**1) Printing On Screen.**

* **Introduction to the print() function in Python.**

In Python, the print() function is used to display output on the screen. It sends the specified message or variable values to the standard output device (usually the monitor). It is one of the most commonly used built-in functions in Python for debugging and showing results.

print("Hello, world!")

* **Formatting outputs using f-strings and format().**

String formatting allows you to create dynamic strings by inserting values into placeholders within a string template. Python provides multiple ways to format strings, each with its own advantages.

**Key Features:**

* **Value Insertion:** Embed variables/expressions directly in strings.
* **Format Specification**:
  + Number formatting (decimal places, padding)
  + Text alignment (left, right , center)
  + Type conversion (decimal, hex, octal)
* **Reuse:** Same value can be used multiple times
* **Dictionary Unpacking**: Use dictionary values directly in formatting

**Types of String Formatting:**

1. f-strings (Formatted String Literals):

* Introduced in Python 3.6
* Most modern and preferred method
* Fast execution (evaluated at runtime)

Example:

name = "Rahul"

age = 20

print(f"My name is {name} and I am {age} years old.")

1. str.format() Method:

* More flexible than %-formatting
* Supports both positional and keyword arguments
* Syntax: "Text {}".format(value)

Example:

name = "uday"

age = 20

print("My name is {} and I am {} years old.".format(name, age))

1. % Formatting (Old Style):

* Similar to C's printf()
* Being phased out (but still seen in older code)

Example:

name = "Om"

age = 20

print("My name is %s and I am %d years old." % (name, age))