Day – 1  
Date – 9th June 2025  
Python Basics

**🔹 What is a Translator?**

A **translator** is a program that converts code written in **high-level programming languages** (like Python, C, Java) into **machine language** (binary code: 0s and 1s) so that a computer can understand and execute it.

There are **two main types** of translators:

* **Compiler**
* **Interpreter**

**🔸 1. Compiler**

**✅ Definition:**

A **compiler** is a translator that **converts the entire high-level source code** into **machine code or bytecode** **at once**, before execution.

**✅ Python Context:**

Python internally uses a compiler to convert code to **bytecode** (not directly machine code), but this is only part of the process.

**🔸 2. Interpreter**

**✅ Definition:**

An **interpreter** is a translator that **converts and executes code line by line**, instead of compiling the whole program at once.

**✅ Python Context:**

Python is **interpreted**. It uses an interpreter (like **CPython**) to convert source code to bytecode and then executes it using the Python Virtual Machine (PVM).

**🔹 Python Translation Process (Behind the Scenes)**

1. **Source Code (.py)**  
   ↓
2. **Compiler** → Converts it to **Bytecode (.pyc)**  
   ↓
3. **Python Virtual Machine (PVM)** → Executes the bytecode line-by-line (interpreted execution)

**🔹 What is a Text Editor?**

A **text editor** is a software application that allows you to **write, edit, and save plain text files**, especially source code in programming languages like Python, C, Java, etc.

For Python programming, text editors are used to **write .py files** and can range from very simple tools to powerful, feature-rich environments.

Examples: Notepad, VScode, PyCharm etc

**🔹 print() Function in Python**

The print() function is one of the most **commonly used built-in functions** in Python. It is used to **display output** (text, variables, results of expressions) on the screen.

**🔹 REPL in terminal**

**REPL** stands for:

* **Read**: Takes the user input (a line of Python code).
* **Eval**: Evaluates or executes the input.
* **Print**: Displays the result/output of the code.
* **Loop**: Repeats the process, waiting for the next command.

**What is REPL?**

It’s an interactive Python shell that allows you to test and execute Python code **line by line**. It's perfect for beginners and developers to experiment and debug quickly.

>>> print("Hello")

Hello

**🧠 Modes of Python Execution**

Python provides **two main modes** for writing and executing code:

**1. Interactive Mode (REPL Mode)  
  
 Open terminal / command prompt**

** Type python or python3 → You’ll see >>> (the prompt)**

2. **Script Mode (File Mode)**

** Save the code in a file, e.g., program.py**

** Run it using:**

**python program.py**

**🧩 sep in Python print() Function**

The sep parameter in the print() function **defines the separator** between multiple values.

print(value1, value2, ..., sep='separator')

**🔹 By Default:**

Python uses a **space ' '** between items.

print("Python", "is", "fun")

# Output: Python is fun

print("2025", "06", "09", sep="-")

# Output: 2025-06-09

**🧩 end in Python print() Function**

The end parameter in the print() function **defines what to print at the end** of the output.

print(value1, value2, ..., end='ending')

print("Hello", end=" ")

print("World")

# Output: Hello World

**🧠 input() Function in Python**

The input() function is used to **take input from the user** as a **string** at runtime.

**📝 Syntax:**

variable = input("Your message to the user: ")

name = input("Enter your name: ")

print("Hello,", name)

**🧠 Variables in Python**

A **variable** is like a **container** or **name** that stores data in your program so you can use it later.

**🔹 Key Point**

* Variables **store values** like numbers, text, etc.
* You **don’t need to declare a data type** (Python is dynamically typed).
* Use **=** to assign a value.

variable\_name = value

**🔠 Case Sensitivity in Python**

Python is a **case-sensitive** programming language.  
This means **uppercase and lowercase letters are treated as different**.

Name = "Harnoor"

name = "Kaur"

print(Name) # Output: Harnoor

print(name) # Output: Kaur

**🆔 id() Function in Python**

The id() function returns the **unique identity** (memory address) of an object in Python.

id(object)

a = 10

b = 10

print(id(a))

print(id(b))

