**Important points about python**

· **Compilation** translates code into binary format but does not run it.

* Compilation errors (syntax errors, type errors, etc.).

· **Execution** runs the compiled (or interpreted) code to produce results.

* Runtime errors (logical errors, memory errors, NameError, ZeroDivisionError, IndexError, AttributeError.etc.).

### ****PVM (Python Virtual Machine)****

PVM (Python Virtual Machine) is an essential component of the Python interpreter that executes Python bytecode. It acts as an **interpreter** that runs the compiled Python code (bytecode) and performs the actual execution.

### ****How PVM Works?****

**Writing the Code**

* 1. You write Python code in a .py file.

**Compilation to Bytecode**

* 1. When you run the Python script, the Python interpreter **compiles** it into **bytecode** (.pyc files in the \_\_pycache\_\_ directory).
  2. Bytecode is an intermediate, low-level representation of the code that is not human-readable.
  3. Bytecode is not machine code

**Execution by PVM**

* 1. The PVM reads the bytecode and executes it step by step.
  2. It translates the bytecode into machine-level instructions for the operating system.
  3. Also call “Runtime Engine”

### ****1. Modular Programming in Python (Modules & Modularity)****

In Python, **modular programming** refers to dividing code into reusable and manageable sections called **modules**.

* A **module** in Python is simply a .py file that contains functions, classes, or variables.
* You can **import** a module into another Python file using the import statement.

### ****What is an Object in Python?****

An **object** is a **real-world entity** that stores **data (attributes)** and **behaviors (methods)**. In Python, **everything is an object**, including numbers, strings, lists, and even functions.

At its core, an **object is a block of memory** that holds information and operations related to a specific type.