

# PyCraft

Building Logic Through Python Programming

Amresh Maurya



# Preface

This book is designed for students to build a strong foundation in Python programming...



# Contents

<b>Preface</b>	<b>iii</b>
0.1 Definition of Python . . . . .	1
0.2 Features of Python . . . . .	1
0.3 Compiler . . . . .	1
0.4 Interpreter . . . . .	1
0.5 Types of Codes and Program Execution . . . . .	2
0.5.1 Source Code . . . . .	2
0.5.2 Machine Code . . . . .	2
0.5.3 Binary Code . . . . .	2
0.5.4 Executable File . . . . .	2
0.5.5 Bytecode . . . . .	2
0.5.6 Intermediate Code . . . . .	2
0.6 Virtual Machine (VM) . . . . .	3
0.7 Python Virtual Machine (PVM) . . . . .	3
<b>Appendix</b>	<b>5</b>
0.8 Python Installation . . . . .	5
<b>Bibliography</b>	<b>7</b>



## 0.1 Definition of Python

Python is a high-level, interpreted programming language that is widely used for general-purpose programming. It emphasizes code readability and allows programmers to write clear and logical programs using a simple and structured syntax.

## 0.2 Features of Python

- **Simple and Readable Syntax** Python uses an English-like syntax, which makes programs easy to read and write.
- **High-Level Language** Python hides complex details of the system, allowing programmers to focus on problem solving.
- **Interpreted Language** Python programs are executed line by line, which makes debugging easier.
- **Object-Oriented** Python supports object-oriented concepts such as classes and objects.
- **Portable (Platform Independent)** Python programs can run on different operating systems without any modification.
- **Extensive Library Support** Python provides a large standard library that supports various programming tasks.
- **Open Source** Python is free to use and distribute.

## 0.3 Compiler

A compiler is a program that translates the entire source code written in a high-level programming language into machine code at once. The generated machine code is saved as an executable file, which can be run later.

**Examples:** C, C++

- Translates the whole program at one time
- Produces an executable file
- Errors are reported after compilation

## 0.4 Interpreter

An interpreter is a program that translates and executes source code line by line. It does not create a separate executable file. Each line is analyzed and executed immediately.

**Examples:** Python, JavaScript

- Translates and executes code line by line
- No separate executable file is generated
- Easier to debug compared to compiled languages

## 0.5 Types of Codes and Program Execution

### 0.5.1 Source Code

**Source Code** is the code that a programmer writes in a high-level programming language like Python, C, or Java. It is human-readable and easy to understand.

**Example:**

```
1 print("Hello, Python!")
```

### 0.5.2 Machine Code

**Machine Code** is the code that the computer's processor can directly understand. It is in binary (0s and 1s) and not readable by humans.

**Example:** 10101010 11001100 (just an illustration)

### 0.5.3 Binary Code

**Binary Code** is the language of computers consisting of 0s and 1s. Machine code is a type of binary code. All programs eventually get converted into binary code so the CPU can execute them.

### 0.5.4 Executable File

**Executable File** is a file that contains machine code ready to be run on a computer. For example, in Windows, programs ending with **.exe** are executable files.

### 0.5.5 Bytecode

**Bytecode** is an intermediate code between source code and machine code. It is generated when a program is compiled but not yet executed by the computer. Python converts source code into bytecode first.

**Example:** .pyc files in Python is Bytecode

### 0.5.6 Intermediate Code

**Intermediate Code** is another name for bytecode. It is not specific to any computer hardware and can be executed by a virtual machine, like Python Virtual Machine or Java Virtual Machine.



## 0.6 Virtual Machine (VM)

A **Virtual Machine (VM)** is a software program that emulates a physical computer. It provides an environment where programs can run as if they were executed on a real computer.

- Acts as a bridge between the program and the hardware.
- Makes programs platform-independent.
- Examples include: Java Virtual Machine (JVM) and Python Virtual Machine (PVM).

## 0.7 Python Virtual Machine (PVM)

The **Python Virtual Machine (PVM)** is a component of the Python system that executes Python bytecode. When you run a Python program:

1. Python source code (.py) is compiled into bytecode (.pyc).
2. The PVM interprets the bytecode and executes it on the computer.

### Key Points:

- Makes Python platform-independent.
- Allows Python programs to run on Windows, Linux, or Mac without modification.
- It is a software interpreter, not hardware.

**Note:** The Python Virtual Machine (PVM) is a software component that interprets Python bytecode and executes it on any platform, making Python programs platform-independent.

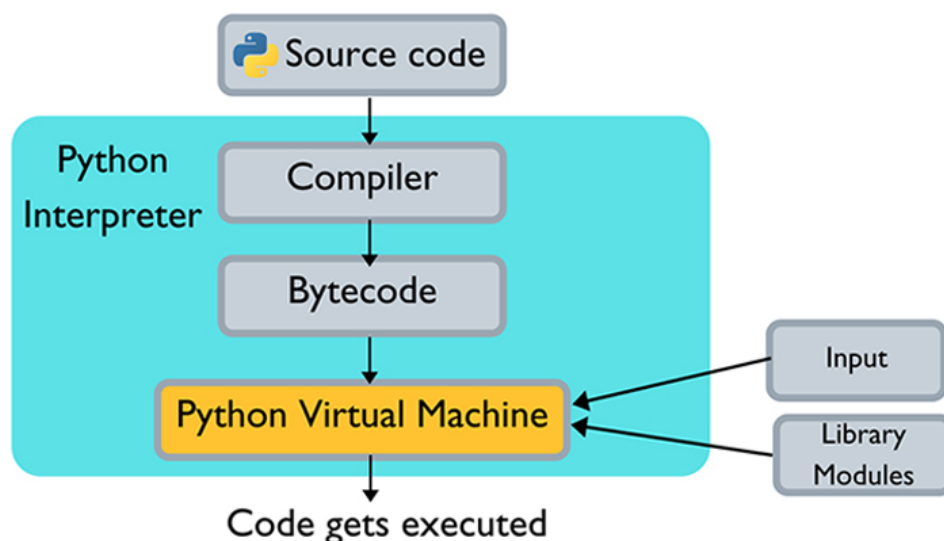


Figure 1: Output of Python Program



# Appendix

## 0.8 Python Installation

Instructions to install Python on Windows, Linux, Mac...



# Bibliography



# Bibliography

- [1] Python Documentation. <https://docs.python.org/3/>
- [2] Leslie Lamport, *LaTeX: A Document Preparation System*.