Unit 1: Introduction to Programming

1. What Are the Fundamentals of Programming?

Fundamentals of Programming refer to the basic building blocks and core principles that are essential to writing computer programs. This includes: understanding how computers execute instructions, writing and organizing code, using variables, data types, control flow, functions, and logic. These principles apply across all programming languages — whether you're writing code in Python, C++, Java, or JavaScript.

2. Why Are We Studying This Course?

This course is important because:
- Programming is the foundation of all modern technology (AI, mobile apps, websites, robots, etc.)
- Learning the fundamentals prepares you to understand any programming language
- It improves logical thinking, problem solving, and creativity
- Programming is a key skill for future careers in technology, science, engineering, and business

At the end of this course, you will be able to:
- Understand how programming languages work
- Write and test simple programs
- Understand the basic tools and environments used by programmers
- Grasp the shared concepts across all programming languages

3. History and Evolution of Programming Languages

A. 1940s–1950s: Machine Language & Assembly (Low-Level Languages)

- Machine Language: Written in binary (0s and 1s), directly understood by computers
- Assembly Language: Uses short keywords (mnemonics) for CPU instructions

Example (Assembly):

```
MOV DX, OFFSET msg
MOV AH, 9
INT 21H
```

These are considered low-level because they are close to the hardware, very fast but hard to write and understand, and not portable.

B. 1960s–1970s: Structured Programming

- Introduced concepts like loops, conditionals, and functions
- Languages: C, Fortran, COBOL

Example (C):

```
#include <stdio.h>
int main() {
  printf("Hello, World!");
  return 0;
}
```

C. 1980s–1990s: Object-Oriented Programming (OOP)

- Introduced concepts like classes, objects, inheritance, and encapsulation
- Languages: C++, Java

Example (C++):

```
class Hello {
 public:
   void sayHello() {
     cout << "Hello, World!" << endl;
   }
};
```

D. 2000s–Today: Modern High-Level Languages

- Focused on simplicity, productivity, and rapid development
- Languages: Python, JavaScript, Ruby, Swift

Example (Python):

```
print("Hello, World!")
```

To Demonstrate Fundamentals of Programming, We Will Use Python

Even though this course is about all programming languages, we will use Python to demonstrate the core concepts because:

- Very easy to read and write
- Excellent for beginners and kids
- Popular in AI, data science, and automation
- Has a huge community and many helpful libraries
- Works across all operating systems (Windows, Linux, macOS)

Brief History of Python

- Created by: Guido van Rossum in 1991
- Inspired by: The comedy show “Monty Python’s Flying Circus”
- Design goal: Make code readable and fun
- Today: Used by Google, NASA, YouTube, Netflix, Facebook, and schools around the world

Setting Up Python

To start coding in Python, you need:

1. Python Interpreter:
- Download from: https://www.python.org
- During installation, check 'Add Python to PATH'

2. Code Editor: Visual Studio Code (VS Code)
- Download from: https://code.visualstudio.com
- Install Python extension from the extensions tab

3. Running Your First Python Program:
- Create a file named hello.py and type:
```
print("Hello, World!")
```
- Run it in the terminal:
```
python hello.py
```