Subject: Computer Skills **Instructor: M.Adnan**

1. High-Level Language

A **high-level language** is a programming language that is closer to human language and abstracted from the details of the computer's hardware.

- ❖ This means that you don't need to know or worry about how the computer works internally (like memory addresses, registers, or how instructions are carried out by the CPU) when you're writing code in a high-level language.

Characteristics:

- ➢ Easier to read, write, and understand
- ➢ More portable (can run on different types of machines)
- ➢ Uses natural language elements (like if, while, print)
- ➢ Requires a compiler or interpreter to translate into machine code

- **Examples:**

Language	Common Use	Example
Python	Web development, data science, automation	`print("Hello, World!")`
Java	Android apps, enterprise software	`System.out.println("Hello");`
C++	Game development, systems programming	`cout << "Hello";`
JavaScript	Web development	`alert("Hello");`

2. Low-Level Language

A **low-level language** is closer to the machine's hardware and less abstracted from the physical instructions the computer executes.

Two Types:

- **Machine Language:** Binary code (e.g., 10100011)
- **Assembly Language:** Symbolic code using mnemonics (e.g., MOV A, B)

Characteristics:

- Harder to read and write
- Faster and more efficient

- Platform-dependent
- Allows direct control over hardware

Examples:

Type	Language	Example
Machine Code	Binary	11001010 00000001
Assembly	x86 Assembly	MOV AX, 1

What is a Compiler?

A **compiler** is a special program that translates code written in a **high-level programming language** (like C, C++, or Java) into **machine code** (also called binary code or executable code) that a computer's hardware can understand and execute.

🧠 **Example:**

C++ Code:

```
#include<iostream>

int main() {

   std::cout << "Hello, World!";

   return 0;

}
```

Advantages of a Compiler:

- **Fast execution** (once compiled)
- **Code optimization** (improves performance)
- **No need to share source code** — just the executable

Disadvantages:

- Takes **time to compile**

- Shows **all errors only after** compilation
- **Less flexible** for quick testing or debugging

What is an Interpreter?

An interpreter is a program that translates and executes code written in a high-level language line by line — without first converting the entire program into machine code like a compiler does.

How an Interpreter Works:

1. **Reads one line** of code.
2. **Translates** it into machine instructions.
3. **Executes** it immediately.
4. Repeats for the **next line**, until the program ends

Example: Python Interpreter