Contents

- **We will learn**
  1. Computer Programming Languages
  - ✓ Machine language
  - ✓ Assembly language
  - ✓ High level language

Computer Programming Languages

Computer Programming Languages:

- A **programming language** is an artificial language that can be used to control the behavior of a machine, particularly a computer.
- Programming languages, like human languages, are defined through the rules, to determine structure and meaning.

3

Computer Programming Languages (Contd…):

- Programming languages are used to facilitate communication about the task of organizing and manipulating information, and to express algorithms precisely.
- For 50 years, computer programmers have been writing codes.
- New technologies continue to emerge, develop, and mature at a rapid pace.
- Now there are more than 2,500 documented programming languages!

4

Most Popular Programming Languages –

C
Python
C++
Java
SCALA
C#
R
Ruby
Go
Swift
JavaScript

5

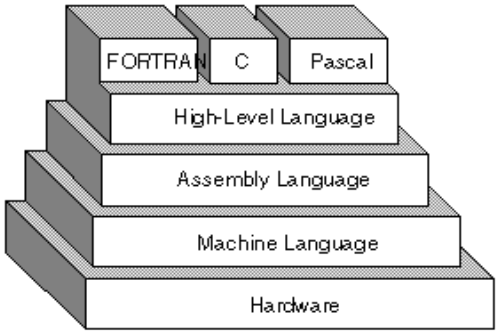
Characteristics of a programming Language –

- ✓ A programming language must be simple, easy to learn and use, have good readability and human recognizable.
- ✓ Programming language's efficiency must be high so that it can be easily converted into a machine code.
- ✓ A programming language should be well structured and documented so that it is suitable for application development.
- ✓ Necessary tools for development, debugging, testing, maintenance of a program must be provided by a programming language.
- ✓ A programming language must be consistent in terms of syntax and semantics.

6

Machine language:

- It is the low-level programming language.



- Machine languages are the only languages understood by computers.

7

Machine language:

- While easily understood by computers, machine languages are almost impossible for humans to use because they consist entirely of numbers.

For example, a processor can execute the following binary instruction as expressed in machine language:

```
Binary: 10110000 01100001
```

8

How these instructions are organized is interesting to note two things.

First, each instruction is composed of a sequence of 1's and 0's. The number of bits that make up a single command vary.

Second, each set of binary digits is interpreted by the CPU into a command to do a very specific job.

However, because different CPUs have different instruction sets, instructions that were written for one CPU type could not be used on another CPU.

9

Assembly Level Language:

- An **assembly language** is a low-level language for programming computers.
- The word "**low**" does not imply that the language is inferior to high-level programming languages but rather refers to the small amount of abstraction between the high level language and machine language.
- It implements a symbolic representation of the numeric machine codes and other constants needed to program a particular CPU architecture.

10

Assembly Level Language (contd):

- An utility program called an **assembler**, is used to translate assembly language statements into the target computer's machine code.

Example: **Assembly language representation is easier to remember**

mov al, 061h

This instruction means:
Move the hexadecimal value 61 into the processor register named "al".

The "mov" is an *operation code* or *opcode*,

A comma-separated list of arguments or parameters follows the opcode;

11

However, assembly still has some downsides.

First, assembly languages still require a lot of instructions to do even simple tasks.

While the individual instructions themselves are somewhat human readable, understanding what an entire program is, can be challenging (it's a bit like trying to understand a sentence by looking at each letter individually).

Second, assembly language still isn't very portable - a program written in assembly for one CPU will likely not work on hardware that uses a different instruction set, and would have to be rewritten or extensively modified.

12

High-level language:

- **High-level languages** are relatively easy to learn because the instructions bear a close resemblance to everyday language.
- The programmer does not require a detailed knowledge of the internal workings of the computer.
- Each instruction in a high-level language is equivalent to several machine-code instructions.
- High-level languages are used to solve problems and are often described as **problem-oriented languages**

13

High-level language (Contd…):

Examples of HLL:

- BASIC was designed to be easily learnt by first-time programmers.
- The language BASIC was an acronym for Beginner's All-Purpose Symbolic Instruction Code.
- It was developed by Dartmouth mathematicians as a teaching tool for undergraduates.
- BASIC was traditionally one of the most commonly used computer programming languages, considered an easy step for students to learn before more powerful languages such as FORTRAN.

14

FORTRAN (formula transition)

- FORTRAN is used for programs solving scientific and mathematical problems.
- It was developed in 1957 by IBM.

15

COBOL

- COBOL is used in solving business problems;
- The word **COBOL** is an acronym that stands for **CO**mmon **B**usiness **O**riented **L**anguage.
- As the expanded acronym indicates, COBOL is designed for developing business, typically file-oriented, applications. It is not designed for writing system programs.

16

Examples of HLL:

- BASIC was designed to be easily learnt by first-time programmers;
- COBOL is used to write programs solving business problems;
- FORTRAN is used for programs solving scientific and mathematical problems.
- With the increasing popularity of windows-based systems, the next generation of programming languages was designed to facilitate the development of GUI interfaces;
- Support for object-oriented programming has also become more common, for example in C++ and Java.