Chapter 2: Programming Language

- Programming Language
-Type of Programming Language
- Generation of Programming Language
- Algorithm and Flowchart

Programming Languages

- Computer program is a set of instructions that makes the computer/hardware to perform a set of particular task in order.
- Without Programs, computers are useless and can do nothing.
- Computers can not understand human natural languages like English Nepali.
- To instruct a computer to perform a certain job , we need a language which can be understand by the computer.
- The language which are used to instruct the computer to do certain jobs are called Computer Programming Languages.
- Example: C,C++, Java, Python etc.

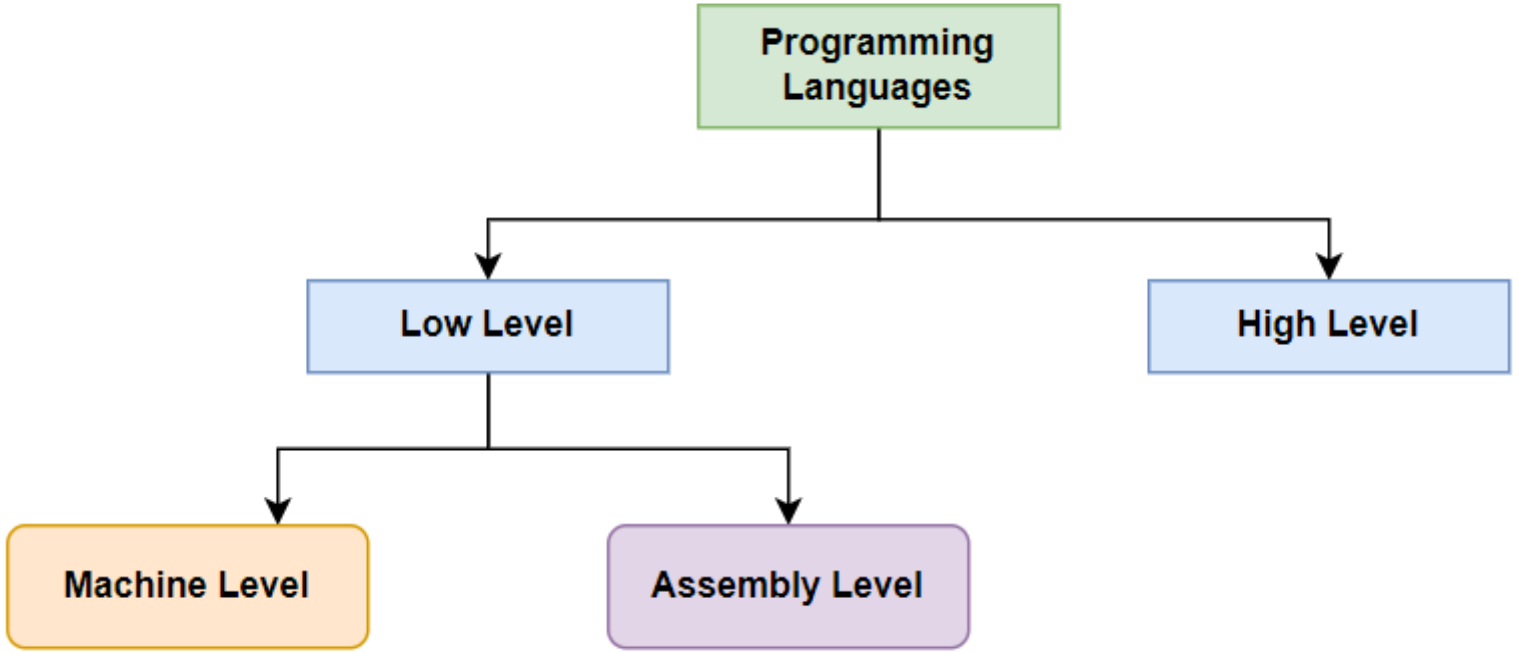
Computer Programming

- Computer programming is the process of writing, testing, debugging/troubleshooting, and maintaining the source code of computer programs.
- This source code is written in a programming language like C++, JAVA, Perl etc.

Computer Programmer

- A **programmer** is someone who writes computer program.
- Computer programmers write, test, and maintain programs or software that tell the computer what to do.

Type/ Level of Programming Languages



Low Level Programming Language

- In Low level programming language each instruction is directly translated into a single machine code.
- It is machine dependent – Only work for a certain machine

1. **Machine Level Language**

- It is computer understandable language
- It consist only binary number (0 and 1)
- Machine language is a sequence of instruction written in the form of binary numbers consisting of 1’s and 0’s to which the computers respond directly.

- The advantage of machine language is that they execute fast as the computer understands them and doesn't need translation into other forms.

Disadvantage:

- It is machine Dependent
- It is Difficult to write
- Example :
- To add two numbers 3 and 6 in machine language like :
- 11 10111001 110
- 3 (machine code for addition) 6
- Difficult to remember different machine codes for different operations.

Assembly Language

- Assembly language is a symbolic representation (called mnemonics)

of machine code.

- The assembly languages program must be translated into machine code by a separate program called an assembler.
- **Assembly Language** → **Assembler** → **Machine Language**
- The assembler program recognizes the character strings that make up the symbolic names of the various machine operations and substitutes the required machine code for each instructions.
- Example:

To add two numbers 3 and 6 in assembly language like :

11 add 110

Sum of the Examples of instructions for assembly language are as follows.

Code	Meaning
ADD	Addition
SUB	Subtraction
INR	Increase
DCR	Decrease
CMP	Compare

Advantage :

- Writing a program in assembly language is more convenient than writing in machine Language
- Program Written in the form of symbolic instruction
- It is useful writing program in embedded system because it needs fewer codes than high level language
- It is also useful for device programming

Disadvantage:

- Hard to Remember mnemonics
- Machine Dependent
- Less efficient than machine language – translated into machine language before execution

High Level Language

- High level programing is a more user friendly and to some extent platform independent.
- It is similar to natural language and so are easy to remember and write.
- High level languages are easy to learn and work but while executing they have to be translated into to machine language.

High level language → **Compiler/Interpreter** → **Machine Language**

- It is slow in execution but efficient for Developing Programs
- C,C++,C#, Java, Python are the example of High level language

1. **Advantages:**

- **Easy to understand and use**
- **Portability-** can be run in different machine with little or no change.
- **Easy Debugging –** Error can be easily detected and removed
- **Fast Development of Software -** Commands of these language are similar closer to English language.

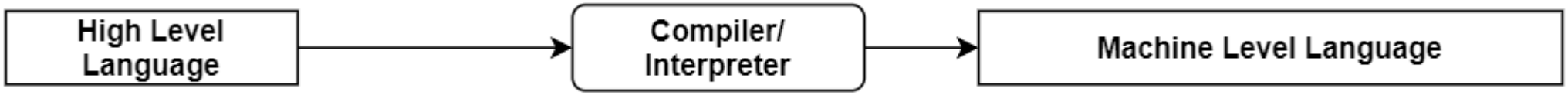
2**. Disadvantage:**

- **It Required more execution time**
- **Need own Translator**

-To change high level language program into machine language , each language has its own translator.

Compiler and Interpreter

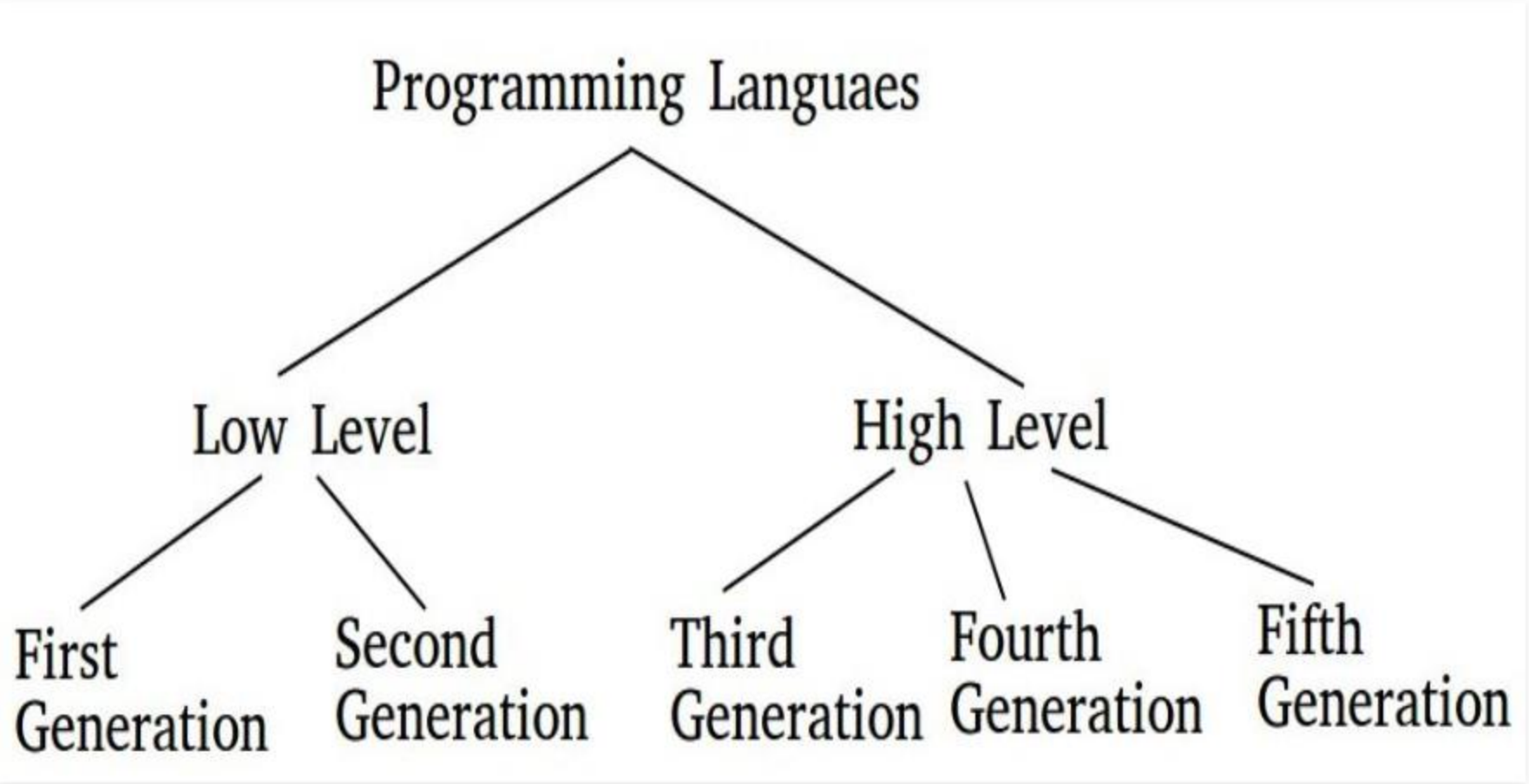
- Compiler and Interpreter are language translator which translate the high level language into a Machine level language.



- The difference between compiler and interpreter is their working Principle.

Compiler	Interpreter
Compiler scans the entire program before translating it into machine code	Interpreter translates and executes the program line by line.
Syntax errors are found only after the compilation of complete program.	Syntax errors can be found after translation of every line.
It takes less execution time	It takes more execution time.
CPU utilization is more.	CPU utilization is less as compared to the compiler.
C, C++, C# etc.	Python, MATLAB

Generation of Programming Language



Generations of Programming Language

- The **first generation languages**, or 1GL, are low-level languages that are machine language.
- The **second generation languages**, or 2GL, are also low-level languages that generally consist of assembly languages.
- The **third generation languages**, or 3GL, are high-level languages such as C.

The **fourth generation languages**, or 4GL, are languages that consist of statements similar to statements in a human language. Fourth generation languages are commonly used in database programming and scripts.

Example : Python, Ruby, SQL

➢**Advantage :**

- Easy to understand and learn
- More User Friendly
- Less time required for application Creation

➢**Disadvantage:**

- Memory consumption is high
- Has poor control over Hardware

- The **fifth-generation languages** are also called 5GL.
- It is based on the concept of artificial intelligence.
- It uses the concept that rather than solving a problem algorithmically, an application can be built to solve it based on some constraints, i.e., we make computers learn to solve any problem.

➢Advantages:

- Machines can make decisions.
- Programmer effort reduces to solve a problem.

➢Disadvantages:

- Complex and long code.
- 2. More resources are required & they are expensive too.

Algorithm and Flowchart

- **Algorithm**

➢ An algorithm is step by step description of the method to solve a problem.

➢ It is an effective procedure for solving a problem in a finite number of steps.

➢ Features:

  ➢ Sequence (Process)

  ➢ Decision (Selection)

  ➢ Repetition (Iteration or loop)

Example : Find the sum of any two numbers

Step 1: Start

Step 2: Read Two Numbers , A and B

Step 3: Sum Two Numbers (C=A+B)

Step 4: Display C as sum of A and B

Step 5: Stop

- **Flowchart**

➢Flowchart is the graphical representation of an algorithm using standard symbols.

➢A flowchart contain a set of various standard shaped boxes that are interconnected by flow lines.

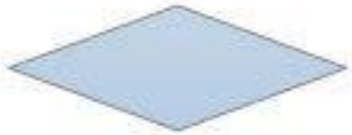
➢Flow lines have arrows to indicate the direction of the flow of control between the boxes.

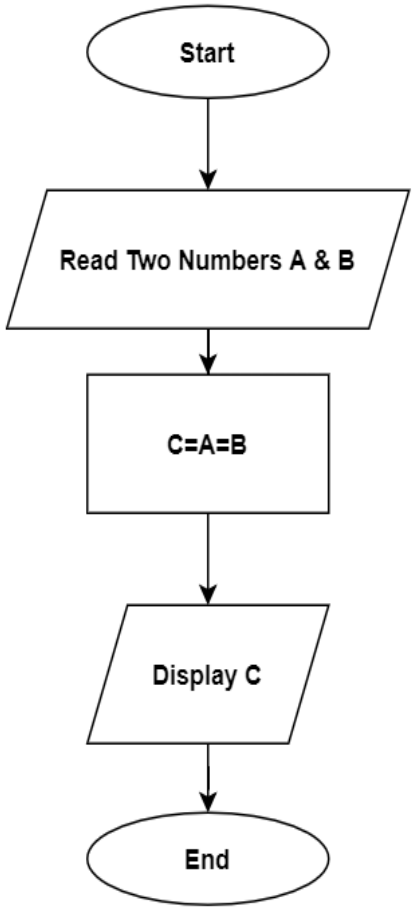
➢Flowcharts facilitate communication between programmers and business persons.

➢Flowcharts help to write the program in any high level language.

➢ Flowcharts provide a clear overview of the entire problem and its algorithm for solution.

Flowchart Symbols

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision



Flowchart for finding the sum of any two Numbers