Programming Introduction

* A computer is an electronic device capable of perfor-ming arithmetic and logical operation.

* A computer system has two components : hardware and software.

* The CPU and the main memory are examples of hardware components.

* In computer system, when a user feed certain data for processing then it is up to the system that is respond on it or not.

* When there is no any action being expressed on it then it converted itself into garbage value.

* When certain processing happens as per the used software then it turns into as "Instruction".

* "The combination of specified instructions is known as Program".

* Every program tells a computer that what to do in order to come up with a solution to a particular problem.

* Programs are written using a programming language.

* The person who writes a program is known as programmer and the way of writing of a program is known as Programming.

* Programming Language -

A programming language is a formal language that is used to communicate instruction to a computer.

* A programming language consists of a set of rules, symbols and syntax that allow programmers to write code that the computer can understand and execute.

Eg:- C, C++, Java, Python, PHP ---

* Programming languages are often classified into different categories -

Here are three main categories of programming language.

i) Low-level languages -

These are programming language that are designed to be used directly with computer hardware.

Eg:- Assembly language, machine language

→ Assemblers are program that translate a program written in assembly language into machine language.

ii) Middle-level language -

These ~~are~~ programming languages are combine elements of both low-level and high-level languages.

Eg:- C and C++

iii) High-level languages -

These programming languages are easy to use and understand. It is designed to be with a focus on code readability and productivity. It is more portable and easier to learn than low-level languages.

Eg:- Java, Python, Ruby, Javascript

* Programming Paradigms -

$\downarrow$

Style of writing programs and codes
(Way of organising the programs)

i) Monolithic Programming
ii) Procedural or Modular Programming
iii) Object Oriented Programming

Monolithic

```
- - - - - -
- - - - - -
- - - - -
- - - - - -
- - - - - -
- - - - - -
- - - - - -
- - - - - -
- - - - -
- - - - - -
```

↓

Everything is in single frame.

Procedural

```
function1()
{
   - - - -
   - - -
   - - - -
}
function2()
{
   - - - - -
   - - - -
   - - - -
}
:
main()
{
   function1()
   function2()
   :
}
```

↓

Every smaller task are divided into function

Object Oriented

```
class info
{
    data1;
    data2;
   function1()
   {
      - - -
      - - -
   }
  function2()
  {
      - - -
      - - -
  }
};
main()
{
    info i;
    i.function1();
    i.function2();
    - - -
    - - -
}
```

↓

In this, we use class and it contains all data and operation together.

* Difference b/w procedural and object oriented programming.

Procedural	Object Oriented
i) Procedural programming is often used for smaller or less complex programs.	i) OOP is often used for large, or more complex programs
ii) It focuses on creating function that operate on data.	ii) It focuses on creating object that contains both data and methods.
iii) It uses top-down approch.	iii) It uses bottom-up approach.
iv) We don't have any access specifiers.	iv) We have access specifiers like public, private, protected etc.
v) It doesn't provide any security.	v) It provides security.
vi) It is difficult to debug and extend any application.	vi) It is easy to debug and extend any application.
vii) Eg:- C, PASCAL	vii) Eg:- Java, C++

* Steps involve in the development and execution of Program -

i) Writing and Editing
ii) Compiling
iii) Linking library
iv) Loading
v) Execution

$\}$ Software that performs all these steps together is known as IDE. (Integrated Development Environment)

IDE - TurboC++, DevCPP, Codeblocks, Xcode, Eclipse, Visual Studio.

- Editing -

It can be done on any text editor / IDE, so that all the above steps can be done at one place.

- Compiling -

Compiler will convert the source code into a machine code, if there are no errors in the program.

- Linking library -

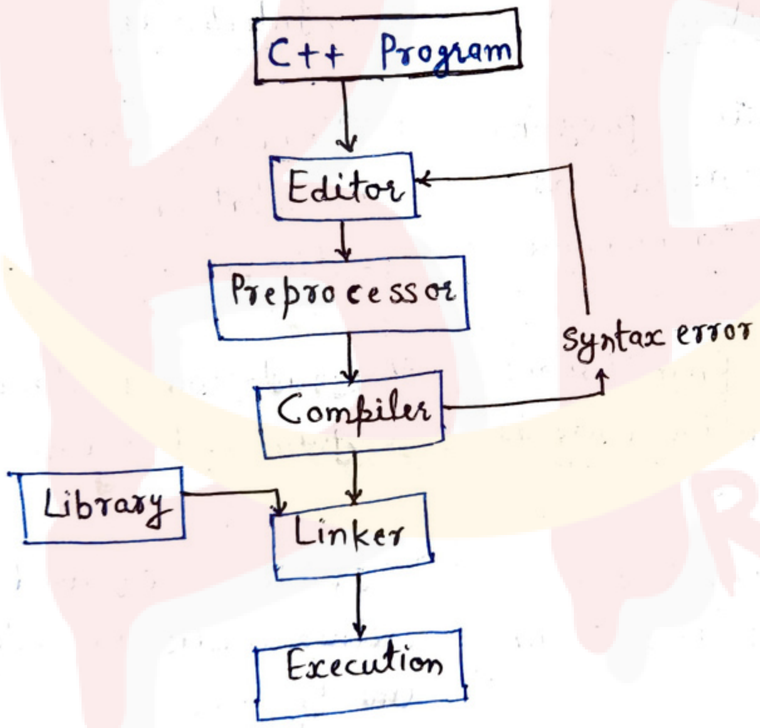
For various operations build in functions/classes are available in C++ that are present in the header files supported by libraries where machine code is readily available to use.

- Loading -

For running the program it should be brought into the main memory from the hard disk for getting it executed this is called loading.

- Execution -

Once the code is in the main memory, OS will ask the CPU to execute the program thus the execution process starts.



Processing of a C++ Program

Compiler v/s Interpreter

The main aim of these two are -

i) To check errors

ii) Convert into machine code

iii) Execution/running a program
(In case of compiler, it does not takes responsibility of execution.)

Compiler	Interpreter
1) Scan the entire program and translate it as a whole into machine code.	1) Translates program one statement at a time.
2) Compilation from source code to machine code is done only once.	2) Traslation is done again and again.
3) As translation is done once then it doesn't run code.	3) It translate and runs/executes the code line by line.
4) Compiler usually take a large amount of time to analyze the source code. However the overall execution time is comparatively faster than interpreter.	4) Interpreters usually take less amount of time to analyze the source code. However the overall execution time is slower than compilers.
5) C, C++, Java use compiler.	6) Javascript, Python, Ruby use interpreters.