Language Processors: Assembler, Compiler and Interpreter

Language Processors –
Assembly language is machine dependent yet mnemonics that are being used to represent instructions in it are not directly understandable by machine and high Level language is machine independent. A computer understands instructions in machine code, i.e. in the form of 0s and 1s. It is a tedious task to write a computer program directly in machine code. The programs are written mostly in high level languages like Java, C++, Python etc. and are called **source code**. These source code cannot be

executed directly by the computer and must be converted into machine language to be executed. Hence, a special translator system software is used to translate the program written in high-level language into machine code is called **Language Processor** and the program after translated into machine code (object program / object code).

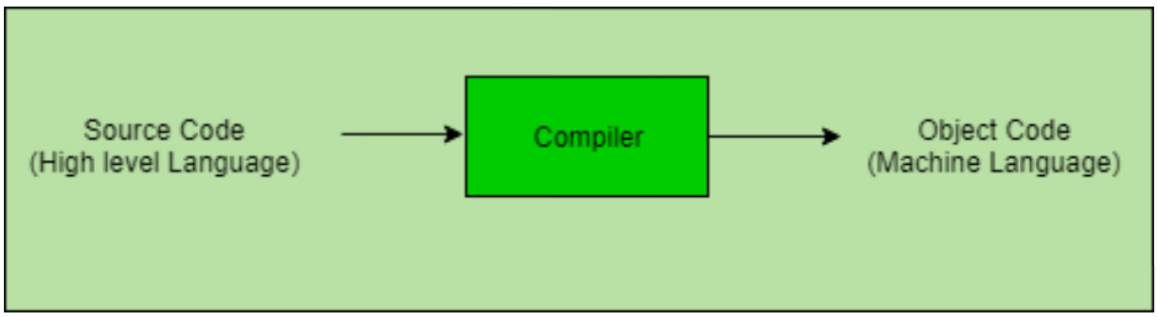
The language processors can be any of the following three types:

1. **Compiler –**

   The language processor that reads the complete source program written in high level language as a whole in one go and translates it into an equivalent program in machine language is called as a Compiler.

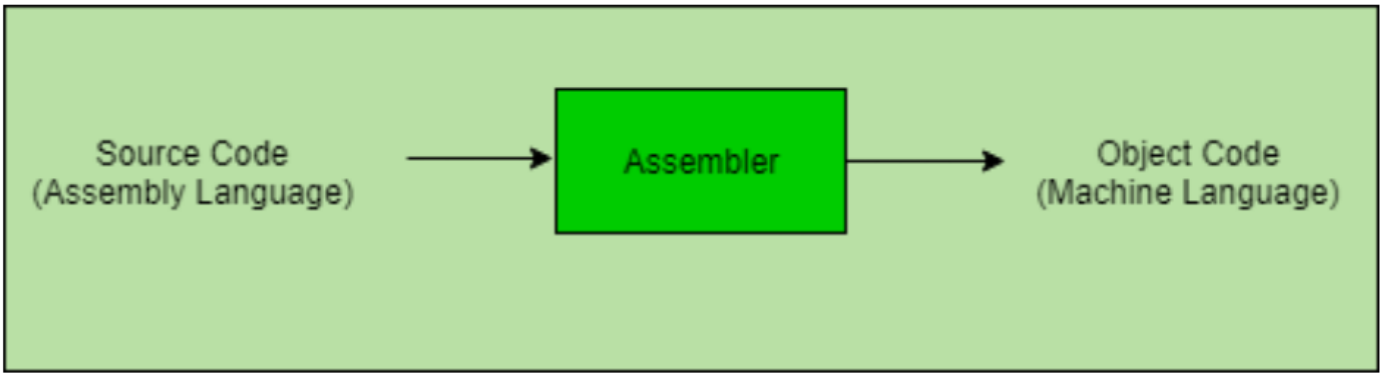
   Example: C, C++, C#, Java

   In a compiler, the source code is translated to object code successfully if it is free of errors. The compiler specifies the errors at the end of compilation with line numbers when there are any errors in the source code. The errors must be removed before the compiler can successfully recompile the source code again.>



2. **Assembler –**

   The Assembler is used to translate the program written in Assembly language into machine code. The source program is a input of assembler that contains assembly language instructions. The output generated by assembler is the object code or machine code understandable by the computer.



3. **Interpreter –**
   The translation of single statement of source program into machine code is done by language processor and executes it immediately before moving on to the next line is called an interpreter. If there is an error in the statement, the interpreter terminates its translating process at that statement and displays an error message. The interpreter moves on to the next line for execution only after removal of the error. An Interpreter directly executes instructions written in a programming or scripting language without previously converting them to an object code or machine code.
   Example: Perl, Python and Matlab.

Difference between Compiler and Interpreter –

Compiler	Interpreter
A compiler is a program which coverts the entire source code of a programming language into executable machine code for a CPU.	interpreter takes a source program and runs it line by line, translating each line as it comes to it.
Compiler takes large amount of time to analyze the entire source code but the overall execution time of the program is comparatively faster.	Interpreter takes less amount of time to analyze the source code but the overall execution time of the program is slower.

Compiler generates the error message only after scanning the whole program, so debugging is comparatively hard as the error can be present any where in the program.	Its Debugging is easier as it continues translating the program until the error is met
Generates intermediate object code.	No intermediate object code is generated.
Examples: C, C++, Java	Examples: Python, Perl