**Covered Topics Under UNIT-1 of "PPS-PROGRAMMING FOR PROBLEM SOLVING (BCS101 / BCS201)"**

shwetatiwari08@recabn.ac.in
shwetatiwari08aug@gmail.com

# Class Notes

*Published Date: November, 2022*

## PPS: UNIT-1

# Introduction to Components of a Computer System

*FALL SEMESTER, YEAR (I/II sem, 1st yr)*

*FALL SESSION (2022-23)*
*(PPS)*
*MS. SHWETA TIWARI*
*Published Date: November, 2022*

[illegible]

shwetatiwari08@recabn.ac.in
shwetatiwari08aug@gmail.com

## TOPIC On : *UNIT-1: Compiler vs Interpreter vs Assembler*

—

By SHWETA TIWARI

**Under On: Introduction to Components of a Computer System**

**PREPARED FOR**
Engineering Students
All Engineering College

**PREPARED BY**
SHWETA TIWARI

## ***Compiler vs Interpreter vs Assembler***

A computer is a combination of hardware and software components. Hardware is just a piece of machine which is controlled by the software's, and they communicate and read only the electrical charge.

That's being said; the software should be written in machine-readable form. Here is where the compiler, interpreter, and Assembler help us. As a programmer, you should be aware of these three utilities and their functions.

## What is a Machine language?

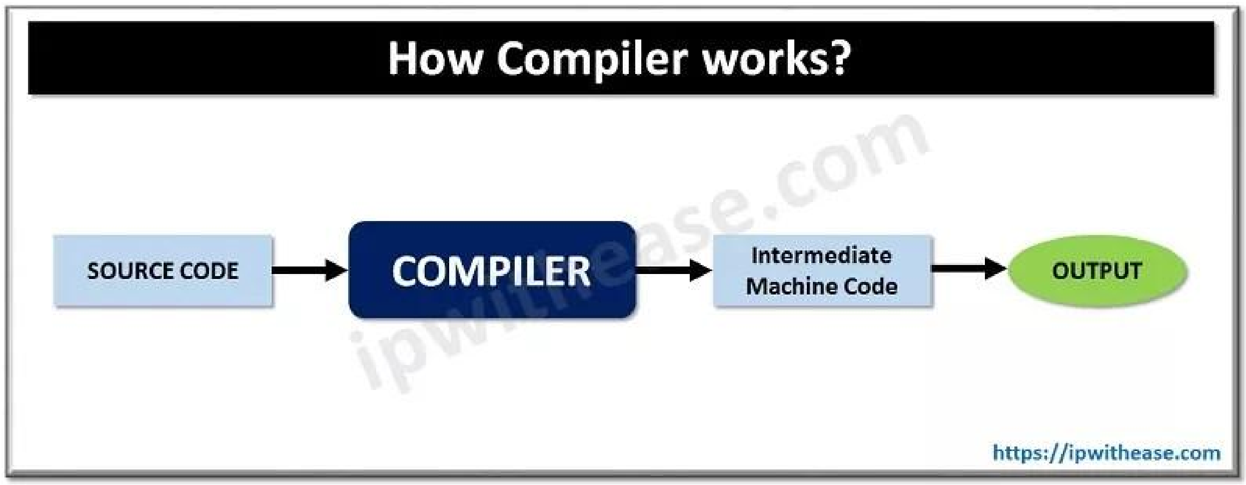
As mentioned earlier the hardware parts of the computers only understand the electrical charge, so the software should be written in the machine language for them to understand. The machine language is called the binary language in software programming.

The binary language consists of two numbers 0 and 1 indicating the power on and off. But writing a program in this language will be hard for a programmer so he writes the program in the high definition languages like C, C++, Python, etc... And use the compiler, interpreter, and assembler to convert them into machine language.

Now let's see each of them in detail-

### What is Compiler?

Compiler converts the high definition programming language into machine-understandable binary codes. It acts more like a translator. Compiler converts the whole code or program into the machine language at a time.



It checks the whole program for errors and displays them. And if there are any syntactic or semantic errors they will be indicated by the compiler. You cannot execute the program without fixing the error.

You can find the compiler in the programming languages such as C, C++. Because of this the execution time for these programs are faster and they are considered the fastest programming language.

## Advantages of Compiler

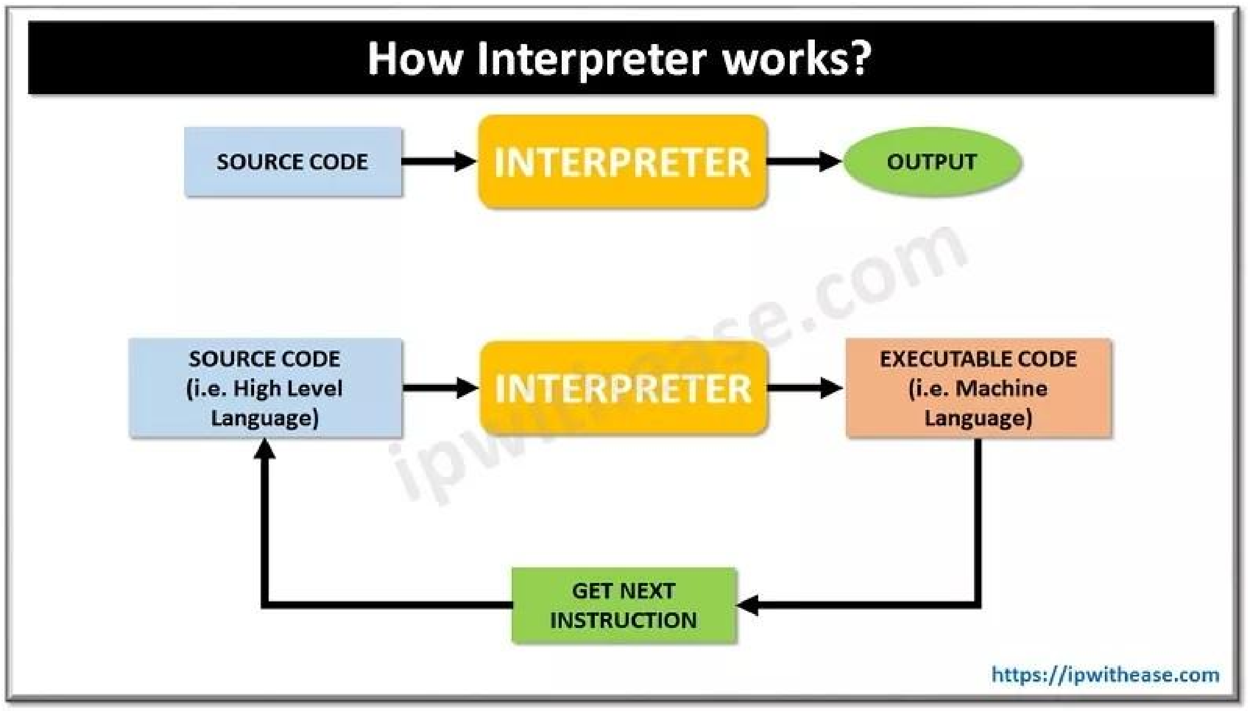
- It produces an executable file that can run without the need for source code.
- It is relatively faster than other language processors.
- Using a compiler is more secure because the actual source program can be hidden which makes it private and secure.
- The machine code of the executable file is native to the machine which makes the compiler well optimized.

## Disadvantages of Compiler

- Debugging(removing errors) is relatively tough.
- It requires a lot of memory for producing an Object File.
- The source code needs to be recompiled every time there is a change in the code.
- After the removal of errors in our code, it needs to be recompiled again.

## What is an Interpreter?

Like the compiler, the interpreter is also a programming language translator which converts the high definition programs into machine-readable codes. But the difference is it converts the program line by line. Thus the scanning time is lower but the overall execution time is higher.



It also shows the error in the program. As it translates or scans only one line at a time, you need to fix the error in the first line for the interpreter to translate the next line. This might take some time, there is no room for errors.

Programming languages such as Python, Ruby, and PHP use the interpreter as a language translator.

## Advantages of Interpreter

- It is easy to find and debug errors from the code/source program.
- As there is no Intermediate object Code, there is less memory consumption.
- Interpreters give execution control to programmers as they can see their code running line by line.
- It can be easily used between different platforms.
- Interpreter is good for fast debugging.

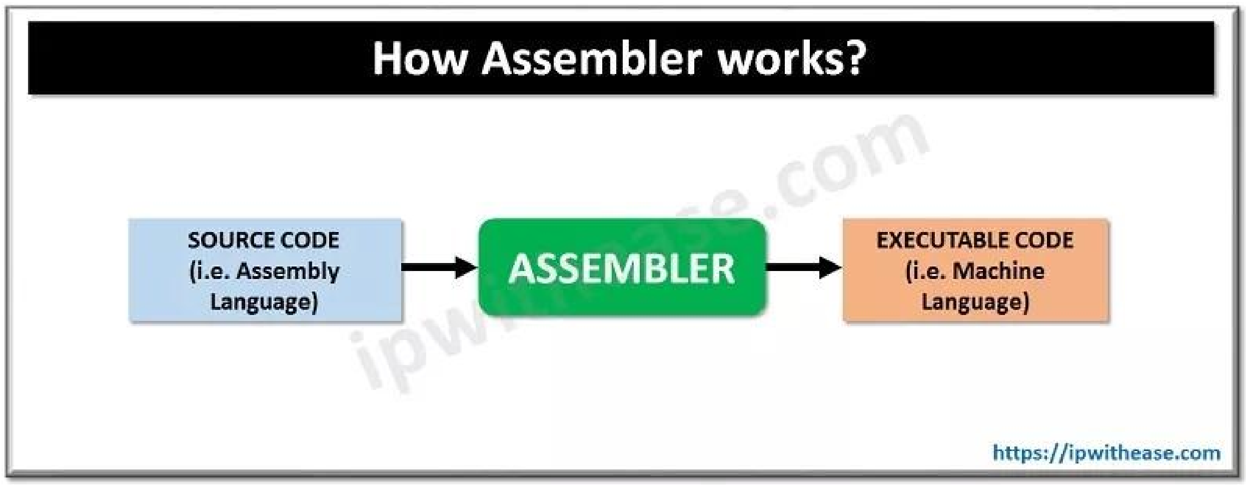
## Disadvantages of Interpreter

- It takes time to convert and execute the instructions line by line.

- It is not good for large programs.
- It is less secure for privacy because we need to share the actual program.
- To run the code on another machine it requires an interpreter to be installed on the machine.

## What is an Assembler?

Other than the high definition languages there is another type of language called assembly language. Assembly language is like an in-between of high definition language and machine languages. It is also known as the low-level language.


How Assembler works?
SOURCE CODE (i.e. Assembly Language)
ASSEMBLER
EXECUTABLE CODE (i.e. Machine Language)
https://ipwithease.com

It is not easily readable by the programmer as it is closer to the machine language than high definition programming languages.  The assembler helps to convert the assembly language into machine codes.

### Advantages of Assembler

- It is a very fast translating system software.
- It is as efficient as the machine language.
- Developing assemblers for translating is easier as compared to compiler and interpreter.

- Assemblers are used in computer forensics, brute force hacking etc. where it is important to determine exactly what is going on.

### Disadvantages of Assembler

- Lower-Level Machine language is difficult to understand and code.
- It changes with the architecture of machines.
- It is difficult to debug.
- We have to write different assembly codes for 8-bit, 16-bit, 32-bit, 64-bit machines which is a tedious task and difficult to maintain.

## Comparison Table: Compiler vs Interpreter vs Assembler:

Here are the major differences between these three types of translators or converters –

| **PARAMETERS** | **COMPILER** | **INTERPRETER** | **ASSEMBLER** |
|---|---|---|---|
| **Conversion** | It converts the high-defined programming language into Machine language or binary code. | It also converts the program-developed code into machine language or binary code. | It converts programs written in the assembly language to the machine language or binary code. |

| | | | |
|---|---|---|---|
| **Scanning** | It scans the entire program before converting it into binary code. | It translates the program line by line to the equivalent machine code. | It converts the source code into the object code then converts it into the machine code. |
| **Error Detection** | Gives the full error report after the whole scan. | Detects error line by line. And stops scanning until the error in the previous line is solved. | It detects errors in the first phase, after fixation the second phase starts. |
| **Code generation** | Intermediate code generation is done in the case of Compiler. | There is no intermediate code generation. | There is an intermediate object code generation. |
| **Execution time** | It takes less execution time compared to an interpreter. | An interpreter takes more execution time than the compiler. | It takes more time than the compiler. |
| **Examples** | C, C#, Java, C++ | Python, Perl, VB, PostScript, LISP, etc… | GAS, GNU |