Difference Between Compiler and Interpreter

Last Updated : 23 Aug, 2024

The Compiler and Interpreter, both have similar works to perform. Interpreters and Compilers convert the Source Code (HLL) to Machine Code (understandable by Computer). In general, computer programs exist in High-Level Language that a human being can easily understand. But computers cannot understand the same high-level language, so for computers, we have to convert them into machine language and make them readable. In this article, we are going to see the differences between them.

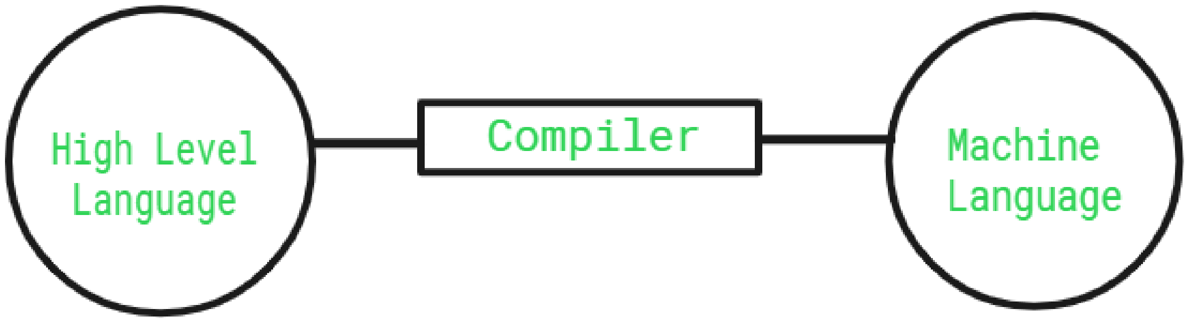
What is a Compiler?

The Compiler is a translator that takes input i.e., High-Level Language, and produces an output of low-level language i.e. machine or assembly language. The work of a Compiler is to transform the codes written in the programming language into machine code (format of 0s and 1s) so that computers can understand.

- A compiler is more intelligent than an assembler it checks all kinds of limits, ranges, errors, etc.
- But its program run time is longer and occupies a larger part of memory. It has a slow speed because a compiler goes through the entire program and then translates the entire program into machine codes.

Role of a Compiler

For Converting the code written in a high-level language into machine-level language so that computers can easily understand, we use a compiler. Converts basically convert high-level language to intermediate assembly language by a compiler and then assembled into machine code by an assembler.



Compiler

Advantages of Compiler

- Compiled code runs faster in comparison to Interpreted code.
- Compilers help improve the security of Applications.
- Compilers give Debugging tools, which help in fixing errors easily.

Disadvantages of Compiler

- The compiler can catch only syntax errors and some semantic errors.
- Compilation can take more time in the case of bulky code.

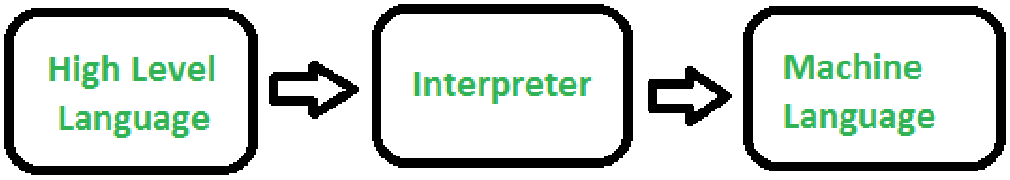
What is an Interpreter?

An Interpreter is a program that translates a programming language into a comprehensible language. The interpreter converts high-level language to an intermediate language. It contains pre-compiled code, source code, etc.

- It translates only one statement of the program at a time.
- Interpreters, more often than not are smaller than compilers.

Role of an Interpreter

The simple role of an interpreter is to translate the material into a target language. An Interpreter works line by line on a code. It also converts high-level language to machine language.



Interpreter

Advantages of Interpreter

- Programs written in an Interpreted language are easier to debug.
- Interpreters allow the management of memory automatically, which reduces memory error risks.
- Interpreted Language is more flexible than a Compiled language.

Disadvantages of Interpreter

- The interpreter can run only the corresponding Interpreted program.
- Interpreted code runs slower in comparison to Compiled code.

Difference Between Compiler and Interpreter

Compiler	Interpreter
Steps of Programming: • Program Creation. • Analysis of language by the compiler and throws errors in case of any incorrect statement. • In case of no error, the Compiler converts the source code to Machine Code. • Linking of various code files into a runnable program. • Finally runs a Program.	**Steps of Programming:** • Program Creation. • Linking of files or generation of Machine Code is not required by Interpreter. • Execution of source statements one by one.
The compiler saves the Machine Language in form of Machine Code on disks.	The Interpreter does not save the Machine Language.
Compiled codes run faster than Interpreter.	Interpreted codes run slower than Compiler.
Linking-Loading Model is the basic working model of the Compiler.	The Interpretation Model is the basic working model of the Interpreter.
The compiler generates an output in the form of (.exe).	The interpreter does not generate any output.
Any change in the source program after the compilation requires recompiling the entire code.	Any change in the source program during the translation does not require retranslation of the entire code.

Compiler	Interpreter
Errors are displayed in Compiler after Compiling together at the current time.	Errors are displayed in every single line.
The compiler can see code upfront which helps in running the code faster because of performing Optimization.	The Interpreter works by line working of Code, that's why Optimization is a little slower compared to Compilers.
It does not require source code for later execution.	It requires source code for later execution.
Execution of the program takes place only after the whole program is compiled.	Execution of the program happens after every line is checked or evaluated.
Compilers more often take a large amount of time for analyzing the source code.	In comparison, Interpreters take less time for analyzing the source code.
CPU utilization is more in the case of a Compiler.	CPU utilization is less in the case of a Interpreter.
The use of Compilers mostly happens in Production Environment.	The use of Interpreters is mostly in Programming and Development Environments.
Object code is permanently saved for future use.	No object code is saved for future use.

Compiler	Interpreter
C, C++, C#, etc are programming languages that are compiler-based.	Python, Ruby, Perl, SNOBOL, MATLAB, etc are programming languages that are interpreter-based.

Conclusion

In summary, compilers and interpreters both serve the purpose of converting high-level code into something a computer can understand, but they do so in different ways. A compiler translates the whole program at once, which can make it run faster but takes more time to compile. An interpreter translates and runs the code line by line, making it easier to catch errors and debug, though it may run slower.

Difference Between Compiler and Interpreter – FAQs

Which is better: Interpreter or Compiler?

The Interpreter is useful in the case of debugging, but it is slower and a Compiler goes for full code, error resolution becomes challenging. Therefore, which one is better, totally depends on what work has to be performed by the user.

Which is faster: Interpreter or Compiler?

Whenever any process is considered, the interpreter is faster than the compiler. But, whenever any program is already compiled, in that case, execution of the compiled program is faster than an interpreted program.

List the types of Compilers?

Here are some types of Compilers listed below:

1. *Cross-Compiler*
2. *Native Compiler*
3. *Bootstrap Compiler*
4. *Decompiler*
5. *Source-to-Source Compiler*
6. *Language Rewriter*
7. *Bytecode Compiler*
8. *Just-in-time Compiler*

List the types of Interpreters?

Here are some types of Interpreters listed below:

1. *Bytecode Interpreter*
2. *Threaded code Interpreter*
3. *Abstract syntax tree Interpreter*
4. *Just-in-time compilation*

imarc...

193

Next Article

Difference Between Transpiler and Compiler

Difference between Native compiler and Cross compiler

1. Native Compiler : Native compiler are compilers that generates code for the same Platform on which it runs. It converts high language into computer's native...

6 min read

Difference Between Assembler and Interpreter

Humans can only understand source codes written in high-level languages like Python, java, c++, etc, or low-level languages like PHP. On the other hand,...

4 min read

Incremental Compiler in Compiler Design

Incremental Compiler is a compiler that generates code for a statement, or group of statements, which is independent of the code generated for other statements....

5 min read

Advantages of Multipass Compiler Over Single Pass Compiler

Programmers, write computer programs that make certain tasks easier for users. This program code is written in High-Level Programming languages like C, C++,...

6 min read

Interpreter Design Pattern in Java

The Interpreter design pattern in Java is a behavioral design pattern that facilitates the interpretation and evaluation of expressions or language grammar...

10 min read

Difference between Cross-Assembler and Compiler

1. Cross-Assembler : A cross-assembler is an assembler that runs on a computer with one type of processor but generates machine code for a different type of...

3 min read

Difference between Compiler and Debugger

1. Compiler :Compiler, as name suggests, is a process that is used to convert code into machine instructions. It simply translates source code from high-level...

Difference Between Transpiler and Compiler

A compiler is a software that converts high-level language to low-level assembly language. A transpiler is another software, sometimes called a source-to-source...

4 min read

Difference between Compiler and Assembler

A compiler and an assembler are significant resources required for the conversion of high-level and low-level programming languages into forms that are...

4 min read

Error detection and Recovery in Compiler

In this phase of compilation, all possible errors made by the user are detected and reported to the user in form of error messages. This process of locating errors a...

6 min read

Article Tags : Compiler Design Difference Between GATE CS