**Difference Between Compiler and Interpreter:**

## **Compilers:**

A compiler is computer software that readily translates programming language into machine code or assembly language or low-level language. It translates every program to binary(1’s and 0’s) that a computer feasibly understands and does the task that corresponds to the code. One condition that a compiler has to follow is the syntax of the programming language that is used.

**Interpreter:**

An interpreter is a computer program that converts program statements into machine code. Program statements include source code, pre-compiled code, and scripts.

## **Difference Between Compiler and Interpreter:**

Computer programs are usually written on high level languages. ... Interpreter translates just one statement of the program at a time into machine code. **Compiler scans the entire program** and translates the whole of it into machine code at once. An interpreter takes very less time to analyze the source code.

While it looks like Compiler and Interpreter work the same by translating programs into machine code, there is a very thin line that differentiates both of them. The differences are as follows:

|  |  |  |
| --- | --- | --- |
| **Basis** | **Compiler** | **Interpreter** |
| Analysis | The entire program is analyzed in a compiler. | Line by line of the program is analyzed in an interpreter. |
| Machine Code | Stores machine code in the disk storage. | Machine code is not stored anywhere. |
| Execution | The execution of the program happens only after the entire program is compiled. | The execution of the program takes place after every line is evaluated and hence the error is raised line by line if any. |
| Run Time | Compiled program runs faster | Interpreted program runs slower. |
| Generation | The compilation gives an output program that runs independently from the source file. | The interpretation does not give any output program and is thus evaluated on every execution. |
| Optimization | The compiler reads the entire program and searches multiple times for a time-saving execution. | No rigorous optimization takes place as code is evaluated line by line |
| Error and error execution | All the errors are shown at the end of the compilation and the program cannot be run until the error is resolved | Displays the errors from line to line. The program runs till the error is found and proceeds further on resolving. |
| Input | The compiler takes in the entire program for analysis. | The interpreter takes in lines of code for analysis. |
| Output | The compiler gives intermediate code forms or object code | The interpreter does not generate any intermediate code forms. |
| Programming languages | C, C++, C#, Java are compiler-based programming languages | PHP, PERL, Ruby are interpreter-based programming languages. |

## **Role of Compiler:**

* Compliers reads the source code, outputs executable code
* Translates software written in a higher-level language into instructions that computer can understand. It converts the text that a programmer writes into a format the CPU can understand.
* **Role of Interpreter:**
* The interpreter converts the source code line-by-line during RUN Time.
* Interpret completely translates a program written in a high-level language into machine level language.
* Interpreter allows evaluation and modification of the program while it is executing.
* Relatively less time spent for analyzing and processing the program
* Program execution is relatively slow compared to compiler

### **Which is faster? Compiler or Interpreter?**

If the process is considered, the Interpreter is faster than the compiler. However, once a program is compiled, Runtime or execution is faster for a compiled program over-interpreted ones.

**MAIN DIFFERENCE:**

The main difference between compiler interpreter and assembler is that **compiler converts the whole high level language program to machine language at a time** while interpreter converts high level language program to machine language line by line and assembler converts assembly language program to machine language.

|  |  |
| --- | --- |
| **Compiler** | **Interpreter** |
| Generates intermediate object code. | No intermediate object code is generated. |
| For Security purpose compiler is more useful. | The interpreter is a little vulnerable in case of security. |
| Examples: C, C++, Java | Examples: Python, Perl, JavaScript, Ruby |
|  |  |

## **KEY DIFFERENCE**

* Compiler transforms code written in a high-level programming language into the machine code, at once, before program runs, whereas an Interpreter coverts each high-level program statement, one by one, into the machine code, during program run.
* Compiled code runs faster while interpreted code runs slower.
* Compiler displays all errors after compilation, on the other hand, the Interpreter displays errors of each line one by one.
* Compiler is based on translation linking-loading model, whereas Interpreter is based on Interpretation Method.
* Compiler takes an entire program whereas the Interpreter takes a single line of code.