**What is a compiler?**

Computers understand only one language and that language consists of sets of instructions made of ones and zeros. This computer language is appropriately called *machine language*.

A single instruction to a computer could look like this:

|  |  |
| --- | --- |
| 00000 | 10011110 |

A particular computer's machine language program that allows a user to input two numbers, adds the two numbers together, and displays the total could include these machine code instructions:

|  |  |
| --- | --- |
| 00000 | 10011110 |
| 00001 | 11110100 |
| 00010 | 10011110 |
| 00011 | 11010100 |
| 00100 | 10111111 |
| 00101 | 00000000 |

As you can imagine, programming a computer directly in machine language using only ones and zeros is very tedious and error prone. To make programming easier, high level languages have been developed. High level programs also make it easier for programmers to inspect and understand each other's programs easier.

This is a portion of code written in C++ that accomplishes the exact same purpose:

|  |  |  |
| --- | --- | --- |
| 1 2 3 4 5 6 7 | int a, b, sum;    cin >> a;  cin >> b;    sum = a + b;  cout << sum << endl; |  |

Even if you cannot really understand the code above, you should be able to appreciate how much easier it will be to program in the C++ language as opposed to machine language.

Because a computer can only understand machine language and humans wish to write in high level languages thence high-level languages have to be re-written (translated) into machine language at some point. This is done by special programs called compilers, interpreters, or assemblers that are built into the various programming applications.

C++ is designed to be a compiled language, meaning that it is generally translated into machine language that can be understood directly by the system, making the generated program highly efficient. For that, a set of tools are needed, known as the development toolchain, whose core are a compiler and its linker.