Program compilation and levels of programming

Canul Cal Jorge Carlos

Robotics 2A

Merida, Yucatan

May 9, 2021

Programming

"Programming is how you get computers to solve problems", (FutureLearn, 2020). Programming is a creative task because there is not only way to solve a problem. In other words, one problem can be solved in different ways. For it, there are choices to make. Another definition could be a set of instructions that are written in a language that the computer can understand (Aman Goel, 2021).

Once taking in consideration these definitions about Programming, it is possible to understand the stages of program compilation and levels of programming.

Stages of program compilation

There are four or six stages depending on the programming language. According to Computer Science in BBC UK (2021), exist six stages starting with lexical analysis, symbol table construction, syntax analysis, semantic analysis, code generation and optimization.

Lexical analysis refers to remove unnecessary comments and spaces to maintain the code clean and easy to understand. As well keywords, constants and identifies are changed by 'tokens' which are symbolic strings; then the symbol table construction is related to build a table in which the names and addresses of all variables should be. As well in this stage, the variables are checked to determine the data types used; after that is the syntax analysis. This step in simple words is checking the tokens created in previous steps to find possible errors and determine the programming language; in the fourth stage of compilation, the variables are checked again to make sure they have been correctly declared and contain the correct data type; in code generation, the machine code is generated; finally, in optimization the main idea is making the program more efficient to run faster and use fewer resources.

In contrast, Calleluks and Javatpoint affirm just there are four stages of compilations which are preprocessing, compilation, assembly and linking. Preprocessing refers to start the code lines with # symbol in order the computer be able to interpretate commands which form a simple macro language with its own syntax and semantics. Compilation is the second stage and when the preprocessed code is translated to assembly instructions specific to the target processor architecture. In simple words, the code is expanded buy the preprocessor and the compiler coverts the code into assembly code (javatpoint, 2018), in the assembly part the assembler is used to assembly instructions to the object code. Finally, in the linking part the code generated before is composed to produce an executable program. The linker will arrange the pieces. At the end, the result of these all processes will generate an executable program.

Levels of programming

To begin with, a programming language defines a set of instructions that are compiled together to perform a specific task. The levels of programming vary depending on the level of abstraction and some programming languages provide low or high level of abstraction. For it, basing on the level of abstractions, the levels of programming are divided in: low-level language and high-level language.

Low-level language.

In simple words, the low-level language programming provides no abstraction to the hardware representing the information in 0 or 1 forms, which are the machine instructions (javatpoint, 2020).

An example of low-level language could be C or C++ because they provide minimal amount of abstraction at the smallest possible cost possible to performance. Despite they have a mid-level of programming, they are considered lower-level languages (ComputerHope, 2019).

High-level language.

The high-level language is a programming language in which are independent of a particular type of computer. The differences between high level and low level languages are that low level language is hard to understand as humans because it is in binary code (0 and 1 forms), but high level is easy to write, read and understand because it maintains English words. As well, a table will be attached to visualize the differences.

Low-level language	High-level language
It is a machine-friendly language, i.e., the computer understands the machine language, which is represented in 0 or 1.	It is a user-friendly language as this language is written in simple English words, which can be easily understood by humans.
The low-level language takes more time to execute.	It executes at a faster pace.
It requires the assembler to convert the assembly code into machine code.	It requires the compiler to convert the high-level language instructions into machine code.
The machine code cannot run on all machines, so it is not a portable language.	The high-level code can run all the platforms, so it is a portable language.
It is memory efficient.	It is less memory efficient.
Debugging and maintenance are not easier in a low-level language.	Debugging and maintenance are easier in a high-level language.

Figure 1. Differences between low-level and high-level languages

Once knowing what high-level language is, the examples of this are Python, Java, C++, C# (even though they can be considered as low level languages), Visual basic and, of course, Javascript because of their facility to understand the code (BBC UK).

References

ComputerHope. (2019). *Low-level language.* USA. ComputerHope. Retrieved from: https://www.computerhope.com/jargon/l/lowlangu.htm#:~:text=A%20low%2Dlevel%20language%20is,are%20assembly%20and%20machine%20code

Computer Science in BBC UK. (2020). *Program Construction.* United Kingdom. BBC UK. Retrieved from: https://www.bbc.co.uk/bitesize/guides/zmthsrd/revision/3

Computer Science in BBC UK. (2020). *Types of programming language.* United Kingdom. BBC UK. Retrieved from: https://www.bbc.co.uk/bitesize/guides/z4cck2p/revision/1

Goel, A. (2021). *What is programming?* India. Hackr.io. Retrieved from: https://hackr.io/blog/what-is-programming

FutureLearn. (2020). *What is programming?* United Kingdom. FutureLearn. Retrieved from: https://www.futurelearn.com/info/courses/programming-101/0/steps/43783

Javatpoint. (2018). *Compilation process in C.* India. Javatpoint. Retrieved from: https://www.javatpoint.com/compilation-process-in-c