Getting Started with C Programming (Part 1): What Is Programming

icodemag.com/getting-started-with-c-programming-part-1-what-is-programming/

Bhupendra Singh

March 25, 2019

Before we begin

This tutorial is the first part of the series “Getting Started with C Programming”. This series will cover the basic concepts of C language with easy explanations and examples. The whole series consists of six modules:

- Part 1: What Is Programming?
- Part 2: First program (Hello World) in C
- Part 3: Operators and expressions
- Part 4: Control statements
- Part 5: Arrays
- Part 6: Functions

Introduction

In the current world of tech, you should also have heard of the term Programming. Programming is the entity which makes every machine to do a specified task. Everything around us is programmed. For example: microwave, washing machines, digital clocks etc.

What is Programming?

First of all, let me introduce you to Programming, and why we program. Okay, simply put, programming is like giving instructions to an electronic device to do something. Say, for example, if we want a machine to do 2+2, we have to instruct it through a program to get our desired result.

Now, lets go a bit deep and discuss how can we program and make our work done by an electronic device (computer here).

How does a computer work?

Alright, as you know that a computer works on electricity. This means that it can only understand electrical signals. Electricity travels in wave form – i.e. crest and trough. A crest represents the maximum value of displacement of a cycle and a trough represents the minimum value of a cycle. In computer science, we represent these high and low values with 1 and 0, called as binary values. 1 refers to ON and 0 refers to OFF.

Machine language

Now, why have I discussed Programming and then how a computer operates? It's because if we want to instruct computer to do something we have to instruct it in binary language, also known as, machine language. That's why in earlier times, programming was done in 0s and 1s. But the issue with that was the complexity of programs. One has to write big complex programs to get even a simpler thing done. Another issue was of removing errors from such a big program.

Later on, to make the things simple, languages like assembly language were introduced. In assembly language, set of particular keywords are there capable of instructing the machine to do something. For example, instead of writing the whole program for addition of two numbers, keyword ADD could be used. In this case, only the operands needs to be supplied along with the keyword ADD to get our result. Example ADD 06H, 05H.

> Note: Here 'H' denotes that the value is in Hexadecimal form.

High-level code

With the passage of time, many languages were introduced. Languages were made to make the programming simple by introducing many human understandable words and syntax. With the use of languages, unlike the old times, one can easily program with the syntax of the language. Code written in human readable form is called as high-level code.

Code translators

But since computers can only understand binary or machine language, special software are used to convert this high-level code to machine-level code. This work is performed by compilers and interpreters. The difference between a compiler and an interpreter is that, a compiler converts the whole written code in one go and reports the errors with line

numbers. The Interpreter, on the other hand, checks the code line by line and if there is any error in the code, it stops there and the program execution cannot be resumed until the error is rectified.

You might also have heard the terms pre-processors, assembler, linker, etc. But that requires a separate explanation of its own explaining the flow/stages of code that it has to go through, to produce the final executable .

Conclusion

Alright, as a conclusion, you write code in high level language, then it goes through some stages, and then you get the desired executable file which can be run directly.

How is it all done?

Now the thing is, how do we do all of this stuff explained above. There are different ways of doing this. Some people just write code in a text editor and compiles it using a compiler manually to make the object file and later on runs the object code. All this can be done through a terminal.

But there’s also something called as Integrated Development Environment abbreviated as IDE. An IDE is a complete package which includes all the stuff you need. It has the compiler as well as a text editor plus much more. Just one click on the button “Build and Run” is required to get our code working.