

# C Language Overview

*This chapter describes the basic details about C programming language, how it emerged, what are strengths of C and why we should use C.*

The C programming language is a general-purpose, high-level language that was originally developed by Dennis M. Ritchie to develop the UNIX operating system at Bell Labs. C was originally first implemented on the DEC PDP-11 computer in 1972.

In 1978, Brian Kernighan and Dennis Ritchie produced the first publicly available description of C, now known as the K&R standard.

The UNIX operating system, the C compiler, and essentially all UNIX applications programs have been written in C. The C has now become a widely used professional language for various reasons.

- Easy to learn
- Structured language
- It produces efficient programs.
- It can handle low-level activities.
- It can be compiled on a variety of computer platforms.

## Facts about C

- C was invented to write an operating system called UNIX.
- C is a successor of B language, which was introduced around 1970.
- The language was formalized in 1988 by the American National Standard Institute. (ANSI).
- The UNIX OS was totally written in C by 1973.

- Today, C is the most widely used and popular System Programming Language.
- Most of the state-of-the-art softwares have been implemented using C.
- Today's most popular Linux OS and RDBMS MySQL have been written in C.

## Why to use C?

C was initially used for system development work, in particular the programs that make up the operating system. C was adopted as a system development language because it produces code that runs nearly as fast as code written in assembly language. Some examples of the use of C might be:

- Operating Systems
- Language Compilers
- Assemblers
- Text Editors
- Print Spoolers
- Network Drivers
- Modern Programs
- Databases
- Language Interpreters
- Utilities

## C Programs

A C program can vary from 3 lines to millions of lines and it should be written into one or more text files with extension ".c"; for example, hello.c. You can use "vi", "vim" or any other text editor to write your C program into a file.

This tutorial assumes that you know how to edit a text file and how to write source code using any programming language.

# C Environment Setup

*This section describes how to set up your system environment before you start doing your programming using C language.*

Before you start doing programming using C programming language, you need the following two softwares available on your computer, (a) Text Editor and (b) The C Compiler.

## Text Editor

This will be used to type your program. Examples of few editors include Windows Notepad, OS Edit command, Brief, Epsilon, EMACS, and vim or vi.

Name and version of text editor can vary on different operating systems. For example, Notepad will be used on Windows, and vim or vi can be used on windows as well as Linux or UNIX.

The files you create with your editor are called source files and contain program source code. The source files for C programs are typically named with the extension ".c".

Before starting your programming, make sure you have one text editor in place and you have enough experience to write a computer program, save it in a file, compile it and finally execute it.

## The C Compiler

The source code written in source file is the human readable source for your program. It needs to be "compiled", to turn into machine language so that your CPU can actually execute the program as per instructions given.

This C programming language compiler will be used to compile your source code into final executable program. I assume you have basic knowledge about a programming language compiler.

Most frequently used and free available compiler is GNU C/C++ compiler, otherwise you can have compilers either from HP or Solaris if you have respective Operating Systems.

Following section guides you on how to install GNU C/C++ compiler on various OS. I'm mentioning C/C++ together because GNU gcc compiler works for both C and C++ programming languages.

## Installation on UNIX/Linux

If you are using Linux or UNIX, then check whether GCC is installed on your system by entering the following command from the command line:

```
$ gcc -v
```

If you have GNU compiler installed on your machine, then it should print a message something as follows:

```
Using built-in specs.  
Target: i386-redhat-linux  
Configured with: ../configure --prefix=/usr .....  
Thread model: posix  
gcc version 4.1.2 20080704 (Red Hat 4.1.2-46)
```

If GCC is not installed, then you will have to install it yourself using the detailed instructions available at <http://gcc.gnu.org/install/>

This tutorial has been written based on Linux and all the given examples have been compiled on Cent OS flavor of Linux system.

## Installation on Mac OS

If you use Mac OS X, the easiest way to obtain GCC is to download the Xcode development environment from Apple's web site and follow the simple installation instructions. Once you have Xcode setup, you will be able to use GNU compiler for C/C++.

Xcode is currently available at [developer.apple.com/technologies/tools/](http://developer.apple.com/technologies/tools/).

## Installation on Windows

To install GCC at Windows you need to install MinGW. To install MinGW, go to the MinGW homepage, [www.mingw.org](http://www.mingw.org), and follow the link to the MinGW download page. Download the latest version of the MinGW installation program, which should be named MinGW-<version>.exe.

While installing MinGW, at a minimum, you must install gcc-core, gcc-g++, binutils, and the MinGW runtime, but you may wish to install more.

Add the bin subdirectory of your MinGW installation to your PATH environment variable, so that you can specify these tools on the command line by their simple names.

When the installation is complete, you will be able to run gcc, g++, ar, ranlib, dlltool, and several other GNU tools from the Windows command line.