

# Learn C Programming in 1 Day

By Krishna Rungta

Copyright 2019 - All Rights Reserved – Krishna Rungta

**ALL RIGHTS RESERVED.** No part of this publication may be reproduced or transmitted in any form whatsoever, electronic, or mechanical, including photocopying, recording, or by any informational storage or retrieval system without express written, dated and signed permission from the author.

# Table Of Content

## [Chapter 1: What is C Programming Language? Basics, Introduction and History](#)

1. [What is C programming?](#)
2. [History of C language](#)
3. [Where is C used? Key Applications](#)
4. [Why learn 'C'?](#)
5. [How 'C' Works?](#)

## [Chapter 2: How to Download & Install GCC Compiler for C in Windows, Linux, Mac](#)

1. [Install C on Windows](#)
2. [Install C in Linux](#)
3. [Install C on MAC](#)

## [Chapter 3: C Hello World! Example: Your First Program](#)

## [Chapter 4: How to write Comments in C Programming](#)

1. [What Is Comment In C Language?](#)
2. [Example Single Line Comment](#)
3. [Example Multi Line Comment](#)
4. [Why do you need comments?](#)

## [Chapter 5: C Tokens, Keywords, Identifiers, Constants, Variables, Data Types](#)

1. [What is a Character set?](#)
2. [Token](#)
3. [Keywords and Identifiers](#)

4. [What is a Variable?](#)
5. [Data types](#)
6. [Integer data type](#)
7. [Floating point data type](#)
8. [Constants](#)

## **Chapter 6: C Conditional Statement: IF, IF Else and Nested IF Else with Example**

1. [What is a Conditional Statement?](#)
2. [If statement](#)
3. [Relational Operators](#)
4. [The If-Else statement](#)
5. [Conditional Expressions](#)
6. [Nested If-else Statements](#)
7. [Nested Else-if statements](#)

## **Chapter 7: C Loops: For, While, Do While, Break, Continue with Example**

1. [What are Loops?](#)
2. [Types of Loops](#)
3. [While Loop](#)
4. [Do-While loop](#)
5. [For loop](#)
6. [Break Statement](#)
7. [Continue Statement](#)
8. [Which loop to Select?](#)

## **Chapter 8: Switch Case Statement in C Programming with Example**

1. [What is a Switch Statement?](#)
2. [Syntax](#)

3. [Flow Chart Diagram of Switch Case](#)
4. [Example](#)
5. [Nested Switch](#)
6. [Why do we need a Switch case?](#)
7. [Rules for switch statement:](#)

## **Chapter 9: C Strings: Declare, Initialize, Read, Print with Example**

1. [What is a String?](#)
2. [Declare and initialize a String](#)
3. [String Input: Read a String](#)
4. [String Output: Print/Display a String](#)
5. [The string library](#)
6. [Converting a String to a Number](#)

## **Chapter 10: Storage Classes in C: auto, extern, static, register with Example**

1. [What is a Storage Class?](#)
2. [Auto storage class](#)
3. [Extern storage class](#)
4. [Static storage class](#)
5. [Register storage class](#)

## **Chapter 11: C Files I/O: Create, Open, Read, Write and Close a File**

1. [How to Create a File](#)
2. [How to Close a file](#)
3. [Writing to a File](#)
4. [Reading data from a File](#)
5. [Interactive File Read and Write with getc and putc](#)

## **Chapter 12: Functions in C Programming with Examples:**

## **Recursive, Inline**

1. [What is a Function?](#)
2. [Library Vs. User-defined Functions](#)
3. [Function Declaration](#)
4. [Function Definition](#)
5. [Function call](#)
6. [Function Arguments](#)
7. [Variable Scope](#)
8. [Static Variables](#)
9. [Recursive Functions](#)
10. [Inline Functions](#)

## **Chapter 13: Pointers in C Programming with Examples**

1. [What is a Pointer?](#)
2. [How does Pointer Work?](#)
3. [Types of a pointer](#)
4. [Direct and Indirect Access Pointers](#)
5. [Pointers Arithmetic](#)
6. [Pointers and Arrays](#)
7. [Pointers and Strings](#)
8. [Advantages of Pointers](#)
9. [Disadvantages of Pointers](#)

## **Chapter 14: Functions Pointers in C Programming with Examples**

## **Chapter 15: C Bitwise Operators: AND, OR, XOR, Shift & Complement (with Example)**

1. [What are Bitwise Operators?](#)
2. [Bitwise AND](#)
3. [Bitwise OR](#)

4. [Bitwise Exclusive OR](#)
5. [Bitwise shift operators](#)
6. [Bitwise complement operator](#)

## **Chapter 16: C Dynamic Memory Allocation using malloc(), calloc(), realloc(), free()**

1. [How Memory Management in C works?](#)
2. [Dynamic memory allocation](#)
3. [The malloc Function](#)
4. [The free Function](#)
5. [The calloc Function](#)
6. [calloc vs. malloc: Key Differences](#)
7. [The realloc Function](#)
8. [Dynamic Arrays](#)

## **Chapter 17: TypeCasting in C: Implicit, Explicit with Example**

1. [What is Typecasting in C?](#)
2. [Implicit type casting](#)
3. [Explicit type casting](#)

# Chapter 1: What is C Programming Language? Basics, Introduction and History

## What is C programming?

C is a general-purpose programming language that is extremely popular, simple and flexible. It is machine-independent, structured programming language which is used extensively in various applications.

C was the basics language to write everything from operating systems (Windows and many others) to complex programs like the Oracle database, Git, Python interpreter and more.

It is said that 'C' is a god's programming language. One can say, C is a base for the programming. If you know 'C,' you can easily grasp the knowledge of the other programming languages that uses the concept of 'C'

It is essential to have a background in computer memory mechanisms because it is an important aspect when dealing with the C programming language.

Language Rank	Types	Spectrum Ranking
1. Python	🌐 🖥️ 📱	100.0
2. C++	📱 🖥️ 📱	99.7
3. Java	🌐 📱 🖥️	97.5
4. C	📱 🖥️ 📱	96.7
5. C#	🌐 📱 🖥️	89.4
6. PHP	🌐	84.9
7. R	🖥️	82.9
8. JavaScript	🌐 📱	82.6
9. Go	🌐 🖥️	76.4
10. Assembly	📱	74.1

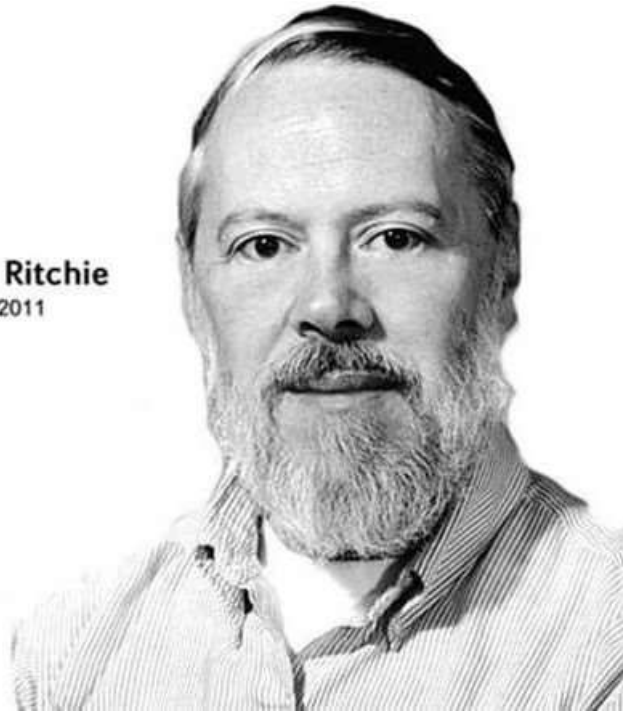
IEEE-the best 10 top programming language in 2018

## History of C language

The base or father of programming languages is 'ALGOL.' It was first introduced in 1960. 'ALGOL' was used on a large basis in European countries. 'ALGOL' introduced the concept of structured programming to the developer community. In 1967, a new computer programming language was announced called as 'BCPL' which stands for Basic Combined Programming Language. BCPL was designed and developed by Martin Richards, especially for writing system software. This was the era of programming languages. Just after three years, in 1970 a new programming language called 'B' was introduced by Ken Thompson that contained multiple features of 'BCPL.' This programming language was created using UNIX operating system at AT&T and Bell Laboratories. Both the 'BCPL' and 'B' were system programming languages.



**Dennis Ritchie**  
1941-2011



In 1972, a great computer scientist Dennis Ritchie created a new programming language called 'C' at the Bell Laboratories. It was created from 'ALGOL', 'BCPL' and 'B' programming languages. 'C' programming language contains all the features of these languages and many more additional concepts that make it unique from other languages.

'C' is a powerful programming language which is strongly associated with the UNIX operating system. Even most of the UNIX operating system is coded in 'C'. Initially 'C' programming was limited to the UNIX operating system, but as it started spreading around the world, it became commercial, and many compilers were released for cross- platform systems. Today 'C' runs under a variety of operating systems and hardware platforms. As it started evolving many different versions of the language were released. At times it became difficult for the developers to keep up with the latest version as the systems were running under the older versions. To assure that 'C' language will remain standard, American National Standards Institute (ANSI) defined a commercial standard for 'C' language in 1989. Later, it was

approved by the International Standards Organization (ISO) in 1990. 'C' programming language is also called as 'ANSI C'.

**ALGOL 1960** 


**BCPL 1967** 

**B 1970** 

**C 1972** 

**K&R C 1978** 

**ANSI C 1989** 

**ANSI/ISO C 1990** 

### History of C

Languages such as C++/Java are developed from 'C'. These languages are widely used in various technologies. Thus, 'C' forms a base for many other languages that are currently in use.

## Where is C used? Key Applications

1. 'C' language is widely used in embedded systems.

2. It is used for developing system applications.
3. It is widely used for developing desktop applications.
4. Most of the applications by Adobe are developed using 'C' programming language.
5. It is used for developing browsers and their extensions. Google's Chromium is built using 'C' programming language.
6. It is used to develop databases. MySQL is the most popular database software which is built using 'C'.
7. It is used in developing an operating system. Operating systems such as Apple's OS X, Microsoft's Windows, and Symbian are developed using 'C' language. It is used for developing desktop as well as mobile phone's operating system.
8. It is used for compiler production.
9. It is widely used in IOT applications.

## Why learn 'C'?

As we studied earlier, 'C' is a base language for many programming languages. So, learning 'C' as the main language will play an important role while studying other programming languages. It shares the same concepts such as data types, operators, control statements and many more. 'C' can be used widely in various applications. It is a simple language and provides faster execution. There are many jobs available for a 'C' developer in the current market.

'C' is a structured programming language in which program is divided into various modules. Each module can be written separately and together it forms a single 'C' program. This structure makes it easy for testing, maintaining and debugging processes.

'C' contains 32 keywords, various data types and a set of powerful built-in functions that make programming very efficient.

Another feature of 'C' programming is that it can extend itself. A 'C' program contains various functions which are part of a library. We can add our features and functions to the library. We can access and use

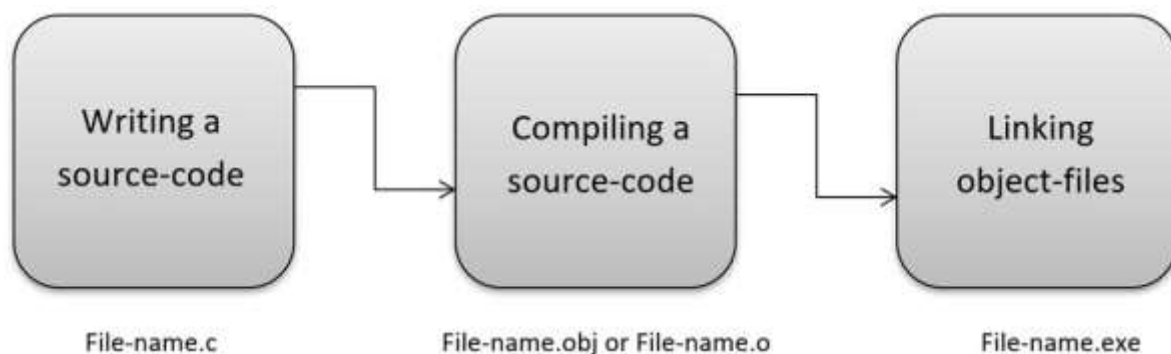
these functions anytime we want in our program. This feature makes it simple while working with complex programming.

Various compilers are available in the market that can be used for executing programs written in this language.

It is a highly portable language which means programs written in 'C' language can run on other machines. This feature is essential if we wish to use or execute the code on another computer.

## How 'C' Works?

C is a compiled language. A compiler is a special tool that compiles the program and converts it into the object file which is machine readable. After the compilation process, the linker will combine different object files and creates a single executable file to run the program. The following diagram shows the execution of a 'C' program



Nowadays, various compilers are available online, and you can use any of those compilers. The functionality will never differ and most of the compilers will provide the features required to execute both 'C' and 'C++' programs.

Following is the list of popular compilers available online:

- Clang compiler
- MinGW compiler (Minimalist GNU for Windows)
- Portable 'C' compiler
- Turbo C

# Summary

- 'C' was developed by Dennis Ritchie in 1972. It is
- a robust language.
- It is a low programming level language close to machine language It is
- widely used in the software development field.
- It is a procedure and structure oriented language.
- It has the full support of various operating systems and hardware platforms.
- Many compilers are available for executing programs written in 'C'.
- A compiler compiles the source file and generates an object file. A
- linker links all the object files together and creates one executable file.
- It is highly portable.

# Chapter 2: How to Download & Install GCC Compiler for C in Windows, Linux, Mac

In this tutorial, we will learn to install C in Windows, Mac, and Linux.

## Install C on Windows

We will use an open-source Integrated Development environment named **Code::Blocks** which bundles a **compiler** (named **gcc** offered by Free Software Foundation GNU), **editor** and **debugger** in a neat package.

**Step 1)** Go to <http://www.codeblocks.org/downloads> and click Binary Release.

**Code::Blocks**

Code

Home Features Downloads Forums Wiki

in

- Home
- Features
- Screenshots
- Downloads
  - Binaries
  - Source
  - SVN
- Plugins
- User manual
- Licensing
- Donations

## Downloads

There are different ways to download and install Code::Blocks on

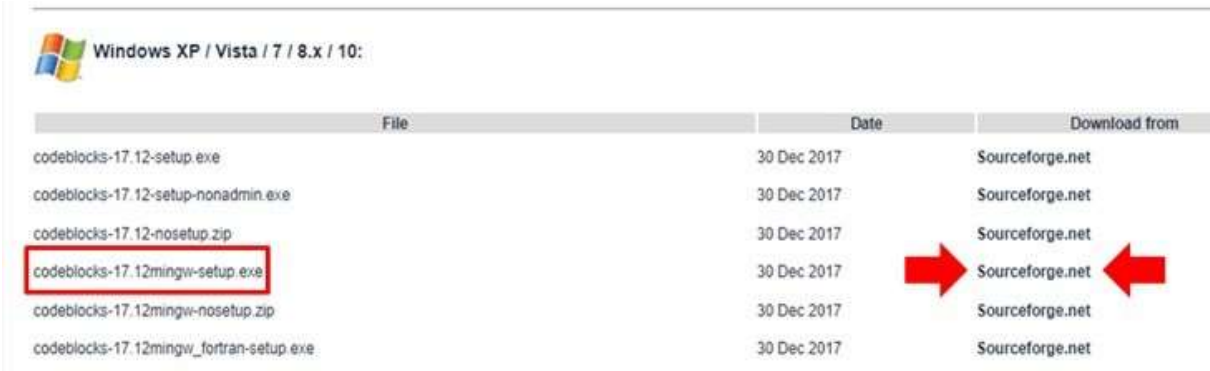
**Download the binary release**

This is the easy way for installing Code::Blocks. Download

- **Download a nightly build:** There are also more re repository. Other distributions usually follow provic
- **Download the source code**

If you feel comfortable building applications from source, t

**Step 2)** Choose the installer with GCC Compiler, e.g., codeblocks-17.12mingw-setup.exe which includes MinGW's GNU GCC compiler and GNU GDB debugger with Code::Blocks source files.

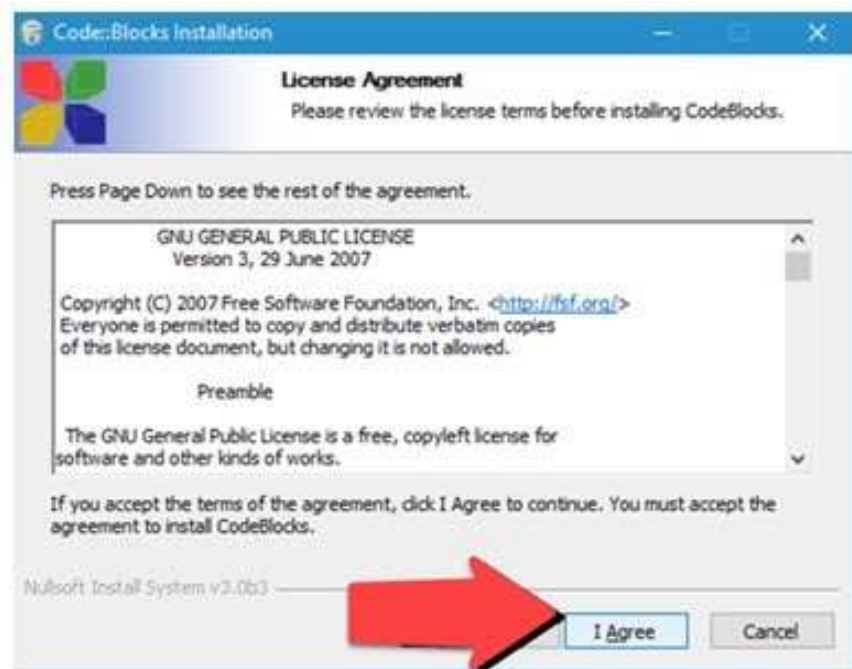


File	Date	Download from
codeblocks-17.12-setup.exe	30 Dec 2017	Sourceforge.net
codeblocks-17.12-setup-nonadmin.exe	30 Dec 2017	Sourceforge.net
codeblocks-17.12-nosetup.zip	30 Dec 2017	Sourceforge.net
<b>codeblocks-17.12mingw-setup.exe</b>	30 Dec 2017	Sourceforge.net
codeblocks-17.12mingw-nosetup.zip	30 Dec 2017	Sourceforge.net
codeblocks-17.12mingw_fortran-setup.exe	30 Dec 2017	Sourceforge.net

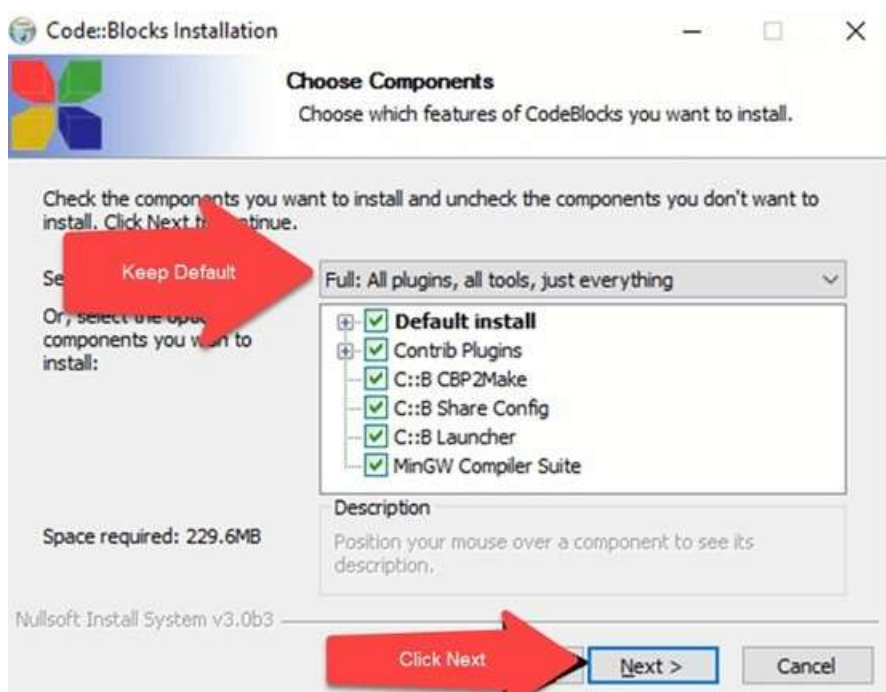
**Step 3)** Run the downloaded installer and accept the default options.



**Step 4)** Accept the Agreement

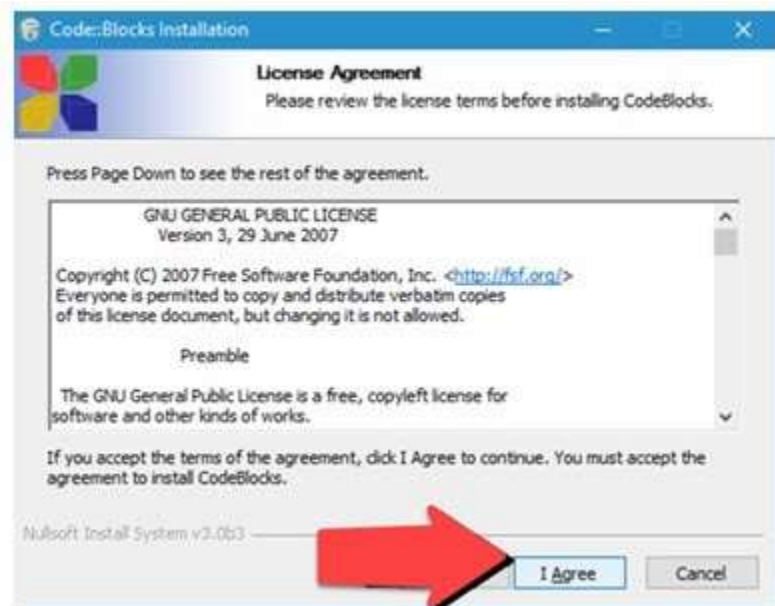


**Step 5)** Keep the component selection default and click Next.



**Step 6)** You may change the installation folder and click Next.

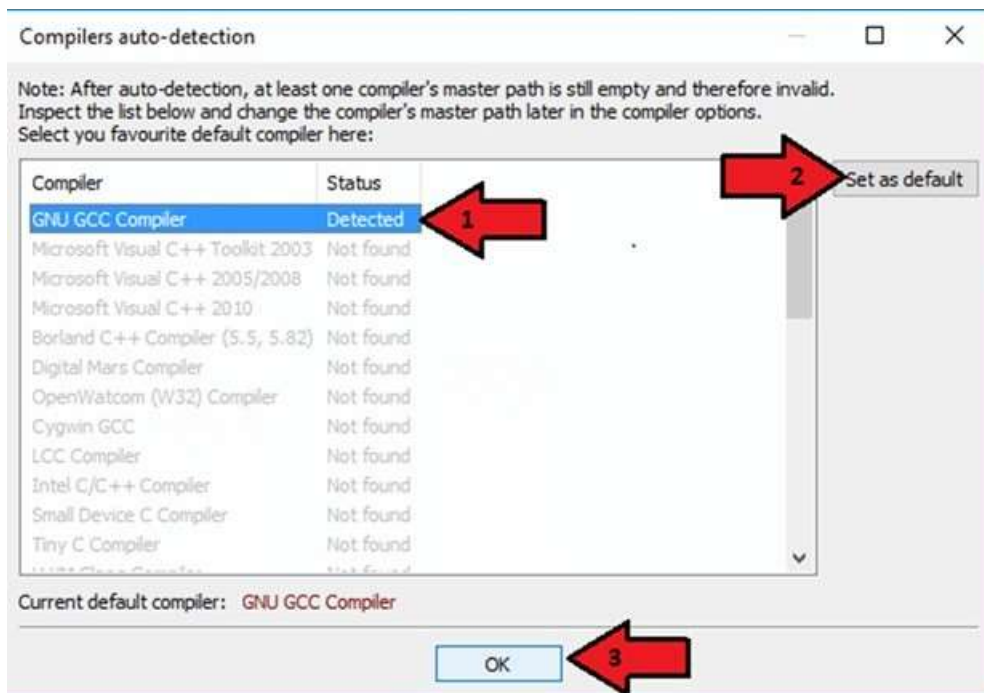




**Step 7)** To launch Code::Blocks double click on the icon.

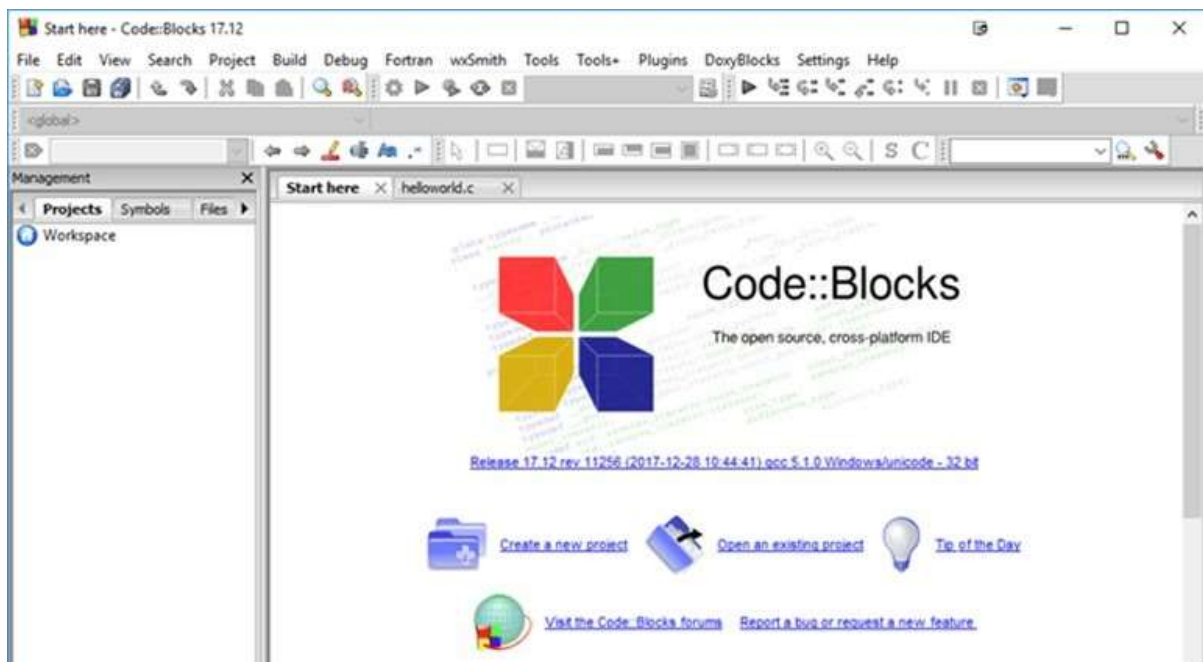


**Step 8)** It will detect the gcc compiler automatically, set it as default.



Associate C/C++ files with code::blocks

**Step 9)** You will see the IDE Home screen.



## Install C in Linux

Linux operating systems mostly comes with GCC preinstalled. To verify if the compiler is installed on the machine, run the following command in the terminal:

```
gcc --version
```

After executing this command if the gcc is installed on the machine then it will return the information about the compiler otherwise it will ask you to install the compiler.

To set up the 'C' environment on Linux distributions follow the given steps:

1. Open terminal.
2. For red-hat, Fedora users, type and execute this command

```
# yum groupinstall 'Development Tools'
```

3. For Debian and Ubuntu users, type and execute following command

```
$ sudo apt-get update
$ sudo apt-get install build-essential manpages-dev
```

4. To verify that the GCC has been successfully installed on the machine as we discussed earlier, execute the following command

```
gcc --version
```

## Install C on MAC

To set up a 'C' programming environment on MAC operating system, follow the given steps:

1. Visit the given link

<https://developer.apple.com/downloads/index.action> and download.

You will need an Apple developer ID

"Command Line Tools for X-Code," pick any version (latest version is always recommended) and download the **.dmg** file.

2. After the file is being downloaded on the machine, double click and follow the wizard and install the file. Always keep the default settings as suggested by the installation wizard.

3. After the installation process, open a terminal and run `gcc -v` command to check if everything is successfully installed.

## Conclusion:

'C' program can be written and executed on any machine that has a suitable environment to run the program. Its recommended using an IDE to run C programs. An IDE includes a compiler, editor and debugger. Clang, MinGW compiler (Minimalist GNU for Windows), Portable 'C' compiler, Turbo C are popular compilers available.

[Buy Now \\$9.99](#)