**What is C language?**

C is actually a computer programming language. It was invented by Dennis Ritchie in early 1970s, in Bell labs. In order to explain further, we can communicate with the computer only by using a special number system called binary , which has only two digits 0,1. Here 0 means no or low voltage and 1 means +5 voltage in digital electronics.   
  
But for a human bein g it is very very difficult( almost impossible) to communicate and write a full program in 0,1. That's why there are something called high level languages, which has English like syntax and can be easily learned by anyone. C is one of those language, which helps us to communicate with the computer , giving it instructions to do something. But as I've said earlier, computer being a digital machine, can only understand 0,1. So, we need some mechanism to translate our C program to 0s and 1s. We call that a compiler.  
  
So, we write a program( some instructions to the computer, to do some specific task ), then compile the program using a compiler, so that the computer can make out what we want to say.

**c**is a programming language which is developed by **Dennis M Ritchie**at Ts bell labs in 1972. c is an structured programming language. It can handle low level activities. it was invented to write an operating system called UNIX. c was a successive language called **“B”** language. c was formalized in 1988 by American national standard institute. c was initially used for system development work .

Why we use “C” language:—→ “C” was adopted as a system development language .

because it generated codes that runs nearly as fast as the code written in assembly language.

**basic structure of a “C” language program :—->**

**Example :**—-> *write a program in “c” language to calculate sum of two numbers.*

*#include<stdio.h>*

*#include<conio.h>*

*void main()*

*{*

*int a, b,c;*

*printf (“Enter first no”);*

*Scanf(“%d”,&a);*

*printf(“Enter Second no”);*

*scanf(“%d”,&b);*

*c=a+b;*

*printf(“the result is”, c);*

*getch()*

*}*

***Example:2:—-> write a program to print the fibonacci Series***

*#include <stdio.h>*

*#include <conio.h>*

*int main()*

*{*

*int n, first = 0, second = 1, next, c;*

*printf("Enter the number of terms\n");*

*scanf("%d", &n);*

*printf("First %d terms of Fibonacci series are:\n", n);*

*for (c = 0; c < n; c++)*

*{*

*if (c <= 1)*

*next = c;*

*else*

*{*

*next = first + second;*

*first = second;*

*second = next;*

*}*

*printf("%d\n", next);*

*}*

*return 0;*

*}*

Mostly for systems programming like writing kernels, operating systems, machine level software.

C++ is used in automobiles, self driving cars too if am not wrong.

Here is a list of areas where c and c++ are used widely.

**Real-World Applications of C**

**1. Operating Systems:**

Scripting of UNIX operating system was the primary purpose behind creation of C. Additionally, as programs scripted in C get executed with speeds equivalent to assembly language, C language has been an integral part of the development of multiple operating systems. Unix-Kernel, Microsoft Windows utilities and operating system applications, and a large segment of the Android operating system have all been scripted in C.

**2. Development of New Languages:**

Efficiency of code execution and simplicity have resulted in C directly or indirectly influencing development of many languages including C++ which is C with classes, C#, D, Java, Limbo, JavaScript, Perl, UNIX’s C Shell, PHP and Python, and Verilog. These languages use C in variable capacity: for instance, in Python, C is used for building standard libraries, while others like C++, Perl and PHP have syntax and control structures based upon C.

**3. Computation Platforms:**

C implements algorithms and data structures swiftly, facilitating faster computations in programs. This has enabled the use of C in applications requiring higher degrees of calculations like MATLAB and Mathematica.

**4. Embedded Systems:**

Various features of C including direct access to machine level hardware APIs, presence of C compilers, deterministic resource use and dynamic memory allocation make C language an optimum choice for scripting applications and drivers of embedded systems.

**5. Graphics and Games:**

C language has been used in the development of a variety of graphics and gaming applications, such as chess, bouncing ball, archery etc.

**Real-World Applications of C++**

**1. Games:**

C++ overrides the complexities of 3D games, optimizes resource management and facilitates multiplayer with networking. The language is extremely fast, allows procedural programming for CPU intensive functions and provides greater control over hardware, because of which it has been widely used in development of gaming engines. For instance, the science fiction game Doom 3 is cited as an example of a game that used C++ well and the Unreal Engine, a suite of game development tools, is written in C++.

**2. Graphic User Interface (GUI) based applications:**

Many highly used applications, such as Image Ready, Adobe Premier, Photoshop and Illustrator, are scripted in C++.

**3. Web Browsers:**

With the introduction of specialized languages such as PHP and Java, the adoption of C++ is limited for scripting of websites and web applications. However, where speed and reliability are required, C++ is still preferred. For instance, a part of Google’s back-end is coded in C++, and the rendering engine of a few open source projects, such as web browser Mozilla Firefox and email client Mozilla Thunderbird, are also scripted in the programming language.

**4. Advance Computations and Graphics:**

C++ provides the means for building applications requiring real-time physical simulations, high-performance image processing, and mobile sensor applications. Maya 3D software, used for integrated 3D modeling, visual effects and animation, is coded in C++.

**5. Database Software:**

C++ and C have been used for scripting MySQL, one of the most popular database management software. The software forms the backbone of a variety of database-based enterprises, such as Google, Wikipedia, Yahoo and YouTube etc.

**6. Operating Systems:**

C++ forms an integral part of many of the prevalent operating systems including Apple’s OS X and various versions of Microsoft Windows, and the erstwhile Symbian mobile OS.

**7. Enterprise Software:**

C++ finds a purpose in banking and trading enterprise applications, such as those deployed by Bloomberg and Reuters. It is also used in development of advanced software, such as flight simulators and radar processing.

**8. Medical and Engineering Applications:**

Many advanced medical equipments, such as MRI machines, use C++ language for scripting their software. It is also part of engineering applications, such as high-end CAD/CAM systems.

**9. Compilers:**

A host of compilers including Apple C++, Bloodshed Dev-C++, Clang C++ and MINGW make use of C++ language.

[C](https://www.invensis.net/it-outsourcing-services/c-plus-plus-application-development-services?utm_campaign=content-link&utm_campaign=blog-post&utm_medium=blog-post&utm_medium=content-link&utm_source=invensis-blog&utm_term=C-plus-plus-in-the-Real-World/?utm_source%3Dinvensis-blog&utm_term=applications-of-c-c-plus-plus-in-the-real-world), C++ for diverse software and platform development requirements, from operating systems to graphic designing applications. Further, these languages have assisted in the development of new languages for special purposes like C#, Java, PHP, Verilog etc. As updating of these languages, particularly C++, continues on a periodic basis, their utilization for robust applications is likely to expand as well.

Language is the way or the medium of sharing our ideas and thoughts. In our real lives we all use certain language to share our ideas.

In the narrow sense, not to go farther i am here sharing the answer to this question to all the readers using English as the language of medium for making the understanding between you and I.

In the broader context, note that here both the readers and writer (you and I) are using English as the mode to share ideas. We both party are known with the English language.

Similarly, in Computing and C programming we all are communicating with the Computer. Computer does what we tell or give the instructions. So as a standard way of communicating with the computer we make use of language, in broader sense, programming language. By specifying the certain instructions following the syntax of the program we tell the computer to perform certain task with the use of the software program.

To enlighten the term, in communication and networking both the message sender and receiver lets say in telephonic conversation, must understand one common language in order to complete the conversation successfully. Both parties must agreed upon certain protocols (lets say understanding same language e.g. English) for making understanding in the entire communication.

C language is the way of making understanding between the computers and the programmers or the users by following certain syntactic rules in order to perform certain tasks. For example, simply addition of two numbers.

C is the high level programming language used for System programming developed by Dennish Ritchie in 1972 while he was working in one of the project with Ken Thomson in AT&T BELL laboratory.

C is the mother of all programming languages that exist today.

C is a structured, procedural programming language that has been widely used both for operating systems and applications and that has had a wide following in the academic community. Many versions of UNIX-based operating systems are written in C. C has been standardized as part of the Portable Operating System Interface (POSIX).

C language is just a high level computer language. High level languages help Human beings to express Logic. Computers dont understand this language. So, you need a mediator. This mediator is usually a ‘Compiler’ or ‘Interpreter’. In case of ‘C’, we have a compiler. The compiler translates the logic expressed in C into Machine Code which is the language of computers. The closest language that one can write that directly translates to Machine code is Assembly language. This language consists of human understandable mnemonics that directly correspond to Machine code.

Consider this piece of logic in C language:

j = 0; i= 0;

for(i=0; i<10; i++) {

j = j+i;

}

This simply calculates the sum of number from 1 to 10. Which we know to be 10\*11/2. But just a naive logic.

In computer terms, a computer can be given instructions of these types:

1. Load/Store from Memory to a CPU register
2. Perform arithmetic and Logic operations on CPU registers (Search for RISC/CISC to know more)
3. Compare CPU registers
4. Jump to step #<some number> based on Comparison

The above C fragment will be compiled to roughly the following instructions

1. Store 0 to MLOC (mem location) of ‘I’
2. Store 0 to MLOC (mem location) of “J”
3. Load MLOC(I) to CPUREG1
4. Compare CPUREG1 with 10
5. IF greater or equal to 10 then JUMP(Step #12)
6. Load MLOC(J) to CPUREG2
7. ADD CPUREG1 to CPUREG2
8. STORE CPUREG2 to MLOC(J)
9. ADD 1 to CPUREG1
10. STORE CPUREG1 to MLOC(J)
11. JUMP (STEP 3)
12. END

Now, it is the compiler that translates the high-level C code to machine-understandable step-by-step operations like the above. As you can see , the following is true:

1. Writing in Machine Code is complicated for Human beings. Compilers can make the job Simple
2. The Generated machine code can be optimized. For example, one can store the Value of J at the end of the loop and cache its value in CPUREG2 during the Loop. (Check “Volatile’ variables in C). Similar case for the COUNTER. It does not need a memory location at all if it is not used later in the code. Compilers do these optimizations to the hilt. It can always be enabled or disabled
3. **C** is the most popular and widely used **programming** **language** which is used for design **computer software**and **applications** that directly interact with **hardware**devices such as *drivers*,*kernels etc*.
4. it was developed in the early 1970s by **“Dennis Ritchie”** at bell laboratories to be used by the **UNIX***operating system*.
5. The main characteristic of **C** **language** that it includes *low-level access* to memory, a simple set of *keywords*, *clean* style and a rich set of *built-in function* and *operators*. Due to this characteristic **C language**is suitable for system programmings like *operating system* and *compiler development.*
6. Many later languages have borrowed directly or indirectly from **C**, including..**C++**, **C#**, S*yntax of***JAVA**, **Python**, **JavaScript**,**PHP...etc.**

C programming language is used for developing system applications that forms a major portion of operating systems such as Windows, UNIX and Linux. Below are some examples of C being used.

* Database systems
* Graphics packages
* Word processors
* Spreadsheets
* Operating system development
* Compilers and Assemblers
* Network drivers
* Interpreters

It is used as a language programming for Unix, several web databases and writing languages for programmers.

295 views · [View 2 Upvoters](https://www.quora.com/What-is-C-language)

[](https://www.quora.com/profile/Wim-ten-Brink)

[Wim ten Brink](https://www.quora.com/profile/Wim-ten-Brink), Officially, COBOL developer. Also done Pascal, C, BASIC and various others.

[Answered Feb 27, 2016](https://www.quora.com/What-is-C-language/answer/Wim-ten-Brink) · Author has **1.3k** answers and **3.9m** answer views

Originally Answered: [What is the C programming language?](https://www.quora.com/What-is-the-C-programming-language?no_redirect=1)

Originally, C was meant to be a programming language to build operating systems with. It was used to create Unix in a way that the whole OS could be ported to different processors without too many changes in the code itself. It has served this purpose for nearly 50 years by now.

However, the C syntax has led to many other different languages that use the same or similar syntax and add more features to the language. Best-known examples are Java and C++, where Java is supposed to compile to machine-independent bytecode and C++ is adding Object-Oriented design patterns.

The C language is meant to be simplistic, yet extremely powerful. And that's till it's most important feature.

C Programming is one of the most basic Programming language which is like the base and foundation of the all the advanced Programming language. To know more about C-Programming and to read the Ultimate guide Notes and Important Question and Answers related to this Topic visit [w3techs - C Programming](http://w3techs.in/Computer-Programming/C-Programming.html)

**5 Ways you can learn Programming Faster.**

**1. Look at the Example Code**

Reading is usually about the words on the page, but learning to program is about code. When you're first learning to program, you should make sure to look at, and try to understand, every example. When I first learned to program, I would sometimes read the code examples before the text, and try to figure out what they did. It doesn't always work, but it did force me to look at the example very carefully, and it often helped make the writeups clearer.

If you want to see what sample code looks like, you can read this site's [introductory programming tutorial](http://w3techs.in/Computer-Programming/C-Programming.html). This tutorial spends a great deal of time talking about the sample code to help you work through exactly what the code does.

**2. Don't Just Read Example Code--Run It**

But when you're reading a programming tutorial (or [book](https://www.cprogramming.com/books.html)), it's easy to look at the sample code and say "I get it, I get it, that makes sense". Of course, you might get it, but you might not get it, and you just don't know it. There's only one way to find out--do something with that code.

If you haven't already, get a [compiler like Code::Blocks](https://www.cprogramming.com/code_blocks/) set up.

Then **type the sample code into a compiler**--if you type it, instead of copying and pasting it, you will really force yourself to go through everything that is there. Typing the code will force you to pay attention to the details of the syntax of the language--things like those funny semicolons that seem to go after every line.

Then compile it and run it. Make sure it does what you think it does.

**Then change it**. Software is the most easily changed machinery on the planet. You can experiment easily, try new things, see what happens; the changes will happen almost immediately, and there is no risk of death or mayhem. The easiest way to learn new language features is to take some code that works one way, and change it.

**3. Write your Own Code as Soon as Possible**

Once you understand something about the language--or even if you're still getting your head around it--start writing sample programs that use it. Sometimes it's hard to find good ideas for what programs to write. That's OK, you don't have to come up with every idea at the beginning.

You can find some [programming challenges](https://www.cprogramming.com/challenge.html) on this site.

You can also reimplement the examples from the book or tutorial you are reading. Try to do so without looking back at the sample code; it won't be as easy as it seems. This technique can work especially well if you tweak the sample code.

If you can't think of a small program to write, but you have in mind a larger program you want to implement, like a game, you could start building small pieces that you can later use for a game. Whether you use them later or not, you will get the same useful experience.

**4. Learn to Use a Debugger**

I already talked about the importance of debugging in [The 5 Most Common Problems New Programmers Face--And How You Can Solve Them](https://www.cprogramming.com/beginner_programming_mistakes.html). But it bears repeating; the sooner you learn good debugging techniques, easier it will be to learn to program.

The first step in doing so is to learn how to use a tool called a [debugger](https://www.cprogramming.com/debuggers.html), which allows you to step through your code.

A debugger will allow you to step line by line through a piece of code. It will let you see the values of variables, and whether the code inside an if statement is executed.

A debugger can help you quickly answer questions about what your code is doing.

1. **int** main()
2. {
3. **int** x;
4. **int** y;
5. **if**( x > 4 ) // <-- what is the value of x here?
6. {
7. y = 5; // <-- did this line of code execute?
8. }
9. }

A final word about debuggers: the first time you learn about a debugger, it will take you longer to fix the problems with your code. After the tenth or so bug, it will really start to pay off. And believe me, you will have way more than ten bugs in your programming career.

I often saw students unwilling to use a debugger. These students really made life hard on themselves, taking ages to find very simple bugs. The sooner you learn to use a debugger, the sooner it will pay off.

**5. Seek out More Sources**

If you don't understand something, there's a good possibility the way it was explained just didn't click.

First, look for alternative explanations. The internet is filled with information about programming, and some explanations work better for different people; you might need pictures, someone else might not. There are also lots of good [books](https://www.cprogramming.com/books.html) with detailed explanations.

But if that doesn't work, the easiest way to figure out where your

