

# Hello World – First C++ Program

Just to give you a little excitement about **C++ programming**, we going to give you a small conventional C++ Hello World program.

C++ is a super set of C programming with additional implementation of object-oriented concepts.

## C++ Program Structure

In this guide, we will write and understand the **first program in C++** programming. We are writing a simple C++ program that prints “Hello World!” message. Let’s see the program first and then we will discuss each and every part of it in detail.

```
#include <iostream>
using namespace std;

// main() is where program execution begins.
int main() {
    cout << "Hello World"; // prints Hello World
    return 0;
}
```

Let us look at the various parts of the above program –

- The C++ language defines several headers, which contain information that is either necessary or useful to your program. For this program, the header **<iostream>** is needed.
- The line **using namespace std;** tells the compiler to use the std namespace. Namespaces are a relatively recent addition to C++.
- The next line **// main() is where program execution begins.** is a single-line comment available in C++. Single-line comments begin with // and stop at the end of the line.
- The line **int main()** is the main function where program execution begins.
- The next line **cout << "Hello World";** causes the message "Hello World" to be displayed on the screen.
- The next line **return 0;** terminates main( )function and causes it to return the value 0 to the calling process.

### Semicolons and Blocks in C++

In C++, the semicolon is a statement terminator. That is, each individual statement must be ended with a semicolon. It indicates the end of one logical entity.

For example, following are three different statements –

```
x = y;  
y = y + 1;  
add(x, y);
```

A block is a set of logically connected statements that are surrounded by opening and closing braces. For example –

```
{  
    cout << "Hello World"; // prints Hello World  
    return 0;  
}
```

C++ does not recognize the end of the line as a terminator. For this reason, it does not matter where you put a statement in a line. For example –

```
x = y;  
y = y + 1;  
add(x, y);
```

is the same as

```
x = y; y = y + 1; add(x, y);
```

### C++ Identifiers

A C++ identifier is a name used to identify a variable, function, class, module, or any other user-defined item. An identifier starts with a letter A to Z or a to z or an underscore (\_) followed by zero or more letters, underscores, and digits (0 to 9).

C++ does not allow punctuation characters such as @, \$, and % within identifiers. C++ is a case-sensitive programming language. Thus, **Manpower** and **manpower** are two different identifiers in C++.

Here are some examples of acceptable identifiers –

```
mohd      zara      abc      move_name  a_123  
myname50  _temp     j        a23b9     retVal
```

## C++ Keywords

The following list shows the reserved words in C++. These reserved words may not be used as constant or variable or any other identifier names.

asm	else	new	this
auto	enum	operator	throw
bool	explicit	private	true
break	export	protected	try
case	extern	public	typedef
catch	false	register	typeid
char	float	reinterpret_cast	typename
class	for	return	union
const	friend	short	unsigned
const_cast	goto	signed	using
continue	if	sizeof	virtual
default	inline	static	void
delete	int	static_cast	volatile
do	long	struct	wchar_t
double	mutable	switch	while
dynamic_cast	namespace	template	

## Whitespace in C++

A line containing only whitespace, possibly with a comment, is known as a blank line, and C++ compiler totally ignores it.

Whitespace is the term used in C++ to describe blanks, tabs, newline characters and comments. Whitespace separates one part of a statement from another and enables the compiler to identify where one element in a statement, such as int, ends and the next element begins.



## C++ Programming

### Statement 1

```
int age;
```

In the above statement there must be at least one whitespace character (usually a space) between `int` and `age` for the compiler to be able to distinguish them.

### Statement 2

```
fruit = apples + oranges; // Get the total fruit
```

In the above statement 2, no whitespace characters are necessary between `fruit` and `=`, or between `=` and `apples`, although you are free to include some if you wish for readability purpose.