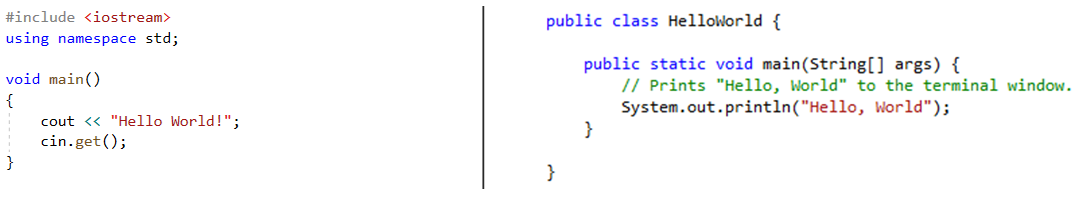
# Introduction to C++ Syntax

## Main

The main function is the starting point of every C++ program. You have to have it in your code somewhere, or the compiler will not know where to start the program from. Inside our HelloWorld project, we have a single file which contains this function. Unlike Java or C#, it is not inside a class, but sits in what we can call ‘*Global Namespace*’. It is basically a function that is on its own in a program.

Here we have a comparison of the HelloWorld program in C++ vs its Java counterpart.



Notice the similarities between the two. Both make use of *void main* to define their starting point, and both have similar construction of functions (*type* -> *name* -> *arguments* -> *code*).

The main difference is that Java has its main function inside of a class, in this case called HelloWorld, whereas C++ has it on its own. C++ and Java both make use of different methods of print their “Hello World” message

## Breaking Down our HelloWorld Program

We are going to break down each line of the C++ program and see what each line of code does.

#### Line 1



The first line asks the compiler to include the library iostream which stands for input output stream. This library gives us functions and classes to retrieve data from the console and output messages to the console. It is the primary method we can use to print the “Hello World!” message. There are other libraries that can manage this, but iostream is a good one to start with.

#### Line 2



The second line is giving us access any definitions within the std namespace. Most classes and functions that are standard to C++ are stored in various headers. Each of these headers generally keeps all its definitions within things called *namespaces*. The use of namespaces is used to provide scope which prevents the program from failing to compile if there are other variables of the same name.

For example, in C++ there is a class called a queue. If we were writing a program and wanted another class called queue, this would conflict with the standard definition as they are both defined in the same space (can’t have two different variables of the same name). Namespaces allow us to define objects in their own named areas. We need to use this as the functions we need to access later (cout and cin) are inside that namespace. Alternatively, items inside a namespace can be accessed by writing the name of the namespace with two colons like so:



#### Line 3



This line tells the compiler that we are defining the main function. We have stated that the main is going to return void (no type or returns nothing), that it takes no arguments (the empty parenthesis).

#### Line 4



Now that we have included the necessary library and started our main function, the next step is to ask the program to print the message. To do this, we use an item called cout which stands for console output.

To push the message to the console, the cout item makes use of the << operator. By adding this on to the code along with our message, we can print whatever we want to the console. Remember as well, as with Java, every line of code in C++ needs semi-colon (;) at the end to signify the end of a command.

#### Line 5



The final line in out code does the opposite of cout. After we have printed the message, we ask the program to wait for input from the console before we can continue. It doesn’t matter what input we receive; we just need a button press to make sure that the program does not close before we have had a chance to view the message.

We then call the .get() function that is part of cin, which asks the program to wait for any console input.

Lastly, we need to close the parenthesis so the code of the main function is enclosed (the program cannot compile otherwise).

## Expanding the Messages

### Chain Link with Cout

Being able to output a single message is fine but there are a couple more useful features we can use to have better control over the cin and cout items in our program.



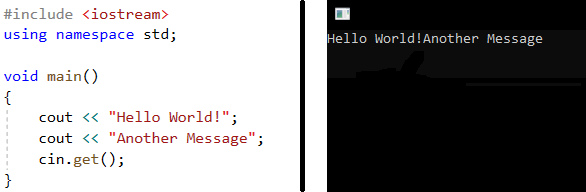
First we can add on more parts to the output by affixing << to the end of each message or command. This allows us to chain link items we want to output to the console. The above example will still print out the message “Hello World!”, but we have split it into multiple parts.



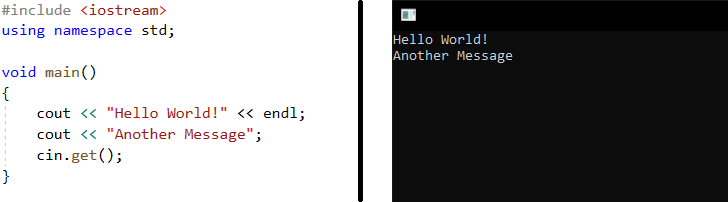
The above example does the same; we have just split the entire message into twelve single characters that we print in one command.

### End Line

It is very useful to be able to print one message on one line of the console, then another message on the next line. Currently, if you used the following code, you would get the following output:



To print the second message on a new line, we can use an item inside the std namespace, endl (stands for end line). It is affixed to the message the same way we have seen above.



### Getting User Input

Sometimes we want to get input from the user, to do that uses the cin mentioned earlier but with a small modification.



## Exercises:

Play about with the HelloWorld program to make sure you are familiar with the console output functions. Remember you can always start a new project and try again if you change something and it no longer compiles. For example:

1. Change the “Hello World!” message to something else.
2. Create a program that takes a single user input and prints it onto the console.
3. Remove Line 2 (using namespace std;) and add the direct namespace reference in front of all the console output and input function calls. E.g.:



1. Create a program that prints the riddle following the formatting:

As I was going to St. Ives,   
I met a man with seven wives.   
Each wife had seven sacks,   
Each sack had seven cats,   
Each cat had seven kits.   
Kits, cats, sacks, and wives,   
How many were going to St. Ives?

## Summary

* You must have the main function somewhere in your C++ code. It cannot be inside a class.
* Make sure you have the correct includes in your code files. For example, without including the iostream library, we cannot use cout or cin.
* If using a standard C++ library (like iostream), we often have use the correct namespace to access these items. State it at the top of the file or use the namespace with each function call as described above.
* Use cout to output a message to the console window.
* Use cin to get input from the console window to your program.
* Use endl to move to the next line in the console window.