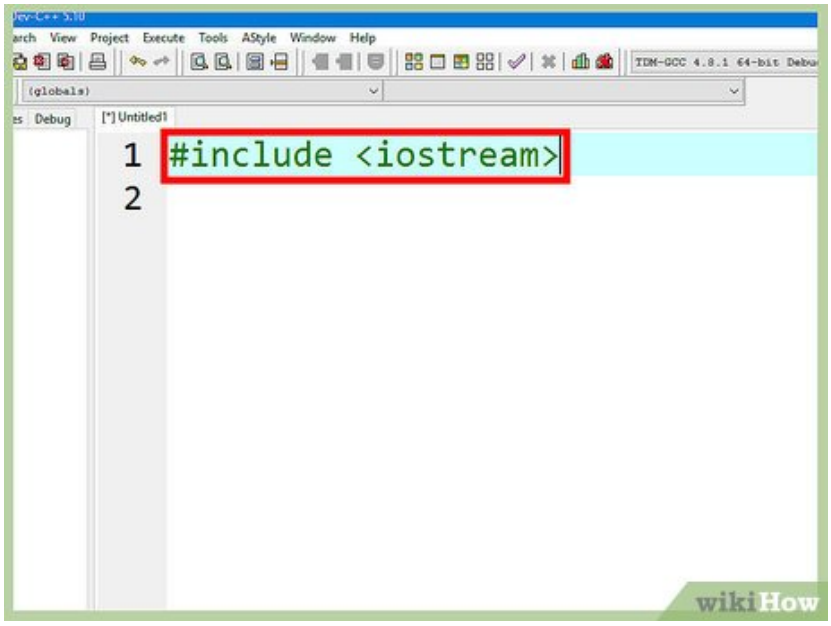
How to Use C++ to Write Cin and Cout Statements

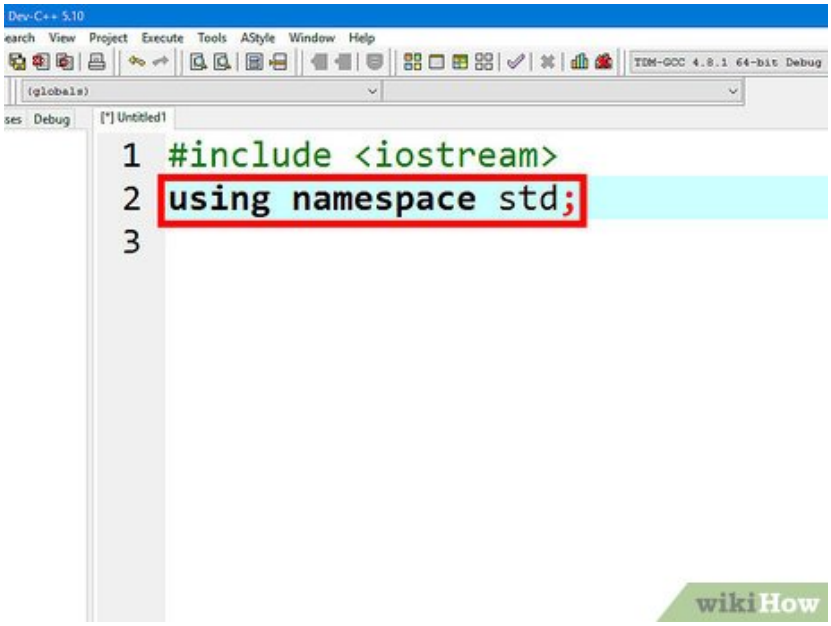
C++ is a very in depth language and can be used for very complex operations, but as with learning any new skill, it is necessary to first learn the fundamentals. This aim of this tutorial is to teach novice programmers how to write simple...

Part 1 of 3:

Setting Up the Main Function

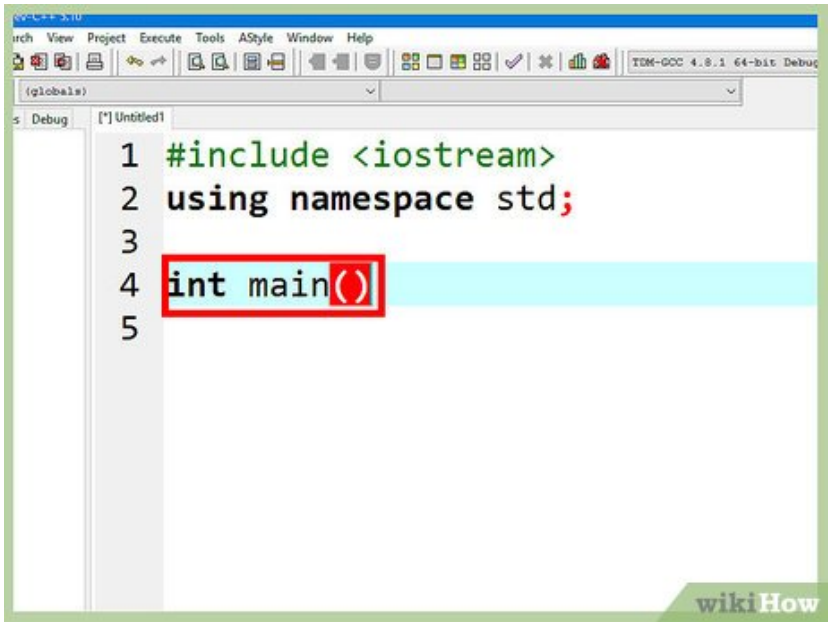
1. 

Include preprocessor directives. These are the first lines of code in the program and are preceded by a hash sign. They are needed for the program to properly compile. In this case the only preprocessor directive needed is iostream, formatted as shown below. Notice that there is no semicolon used at the end of this statement.

2. 

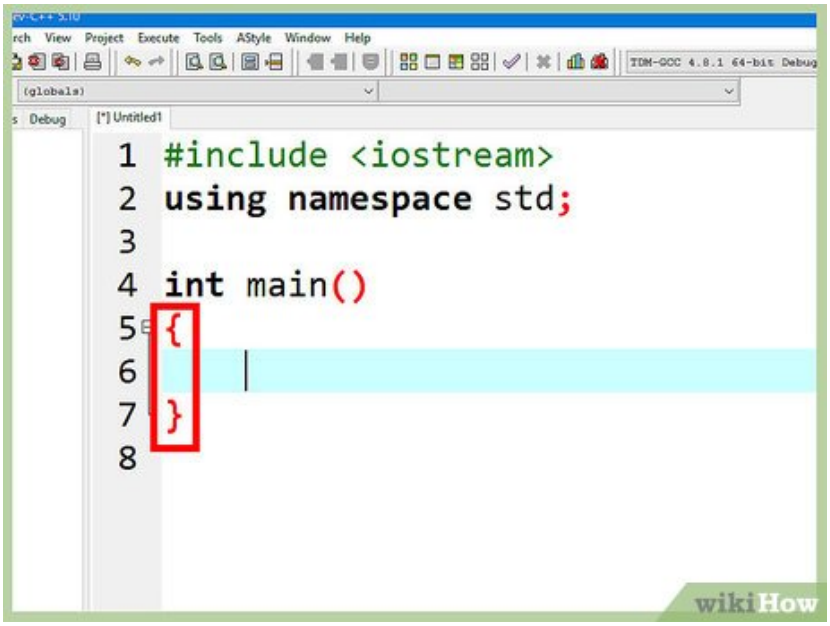
Use standard namespace. In addition to preprocessor directives, the first lines of the code must also define the namespace being used. The standard namespace, formatted as shown below, is sufficient for this code. Note that this line ends with a semicolon.

As a side note, the "using namespace std" line is known as an using directive. Using directives are considered bad practice, as uses of them increases the chances of naming collisions. If you don't want to use using directives, simply prefix every standard library feature with "std::". For example, cout -> std::cout and cin -> std::cin. This works for nested namespaces too, so ios::out would be std::ios::out, numeric_limits::max() would be std::numeric_limits::max().

1. 

Define the main function. To create the main function, type 'int main()' as shown below. The parentheses are for setting the parameters of the function, but here no parameters are needed and thus the

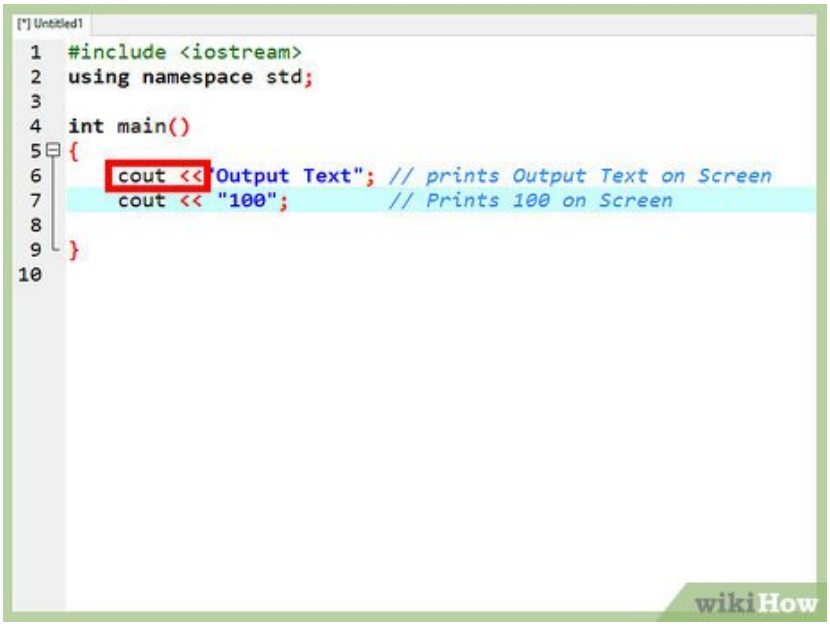
parentheses are empty. There is no semicolon after the function definition.

2. 

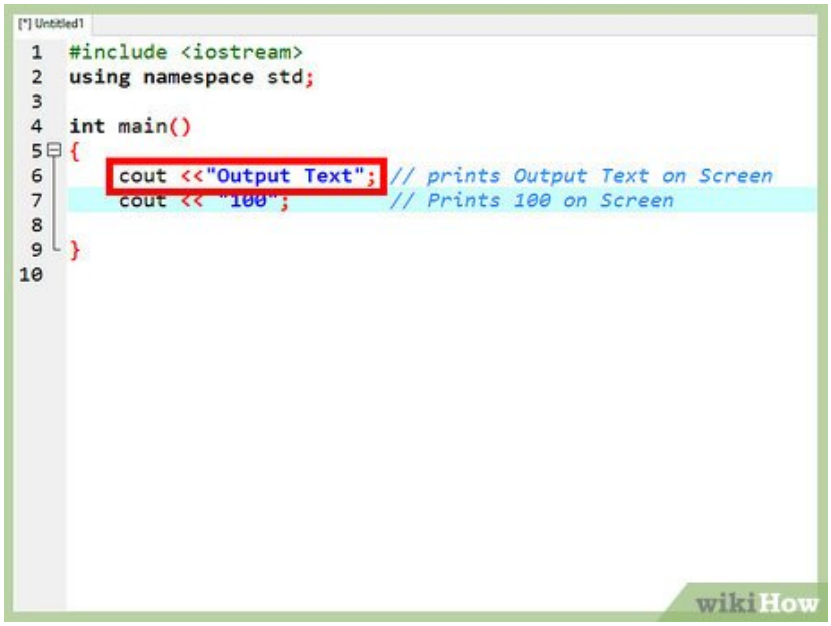
Make curly braces immediately following the function. On the next line, make a set of curly braces as shown in the graphic. Everything included within these curly brackets is part of the main function. The code up to this point should look something like the picture below.

Part 2 of 3:

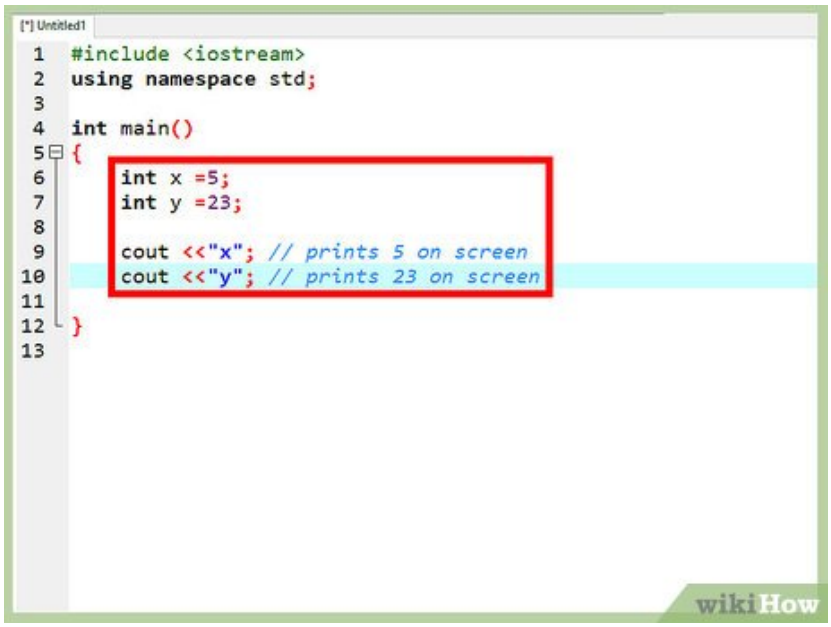
Writing Cout Statements

1. 

Know the syntax. Cout is used with the insertion operator, which is written as (two 'less than' signs). The actual output then follows, written within quotation marks. The line must end with a semicolon.

2. 

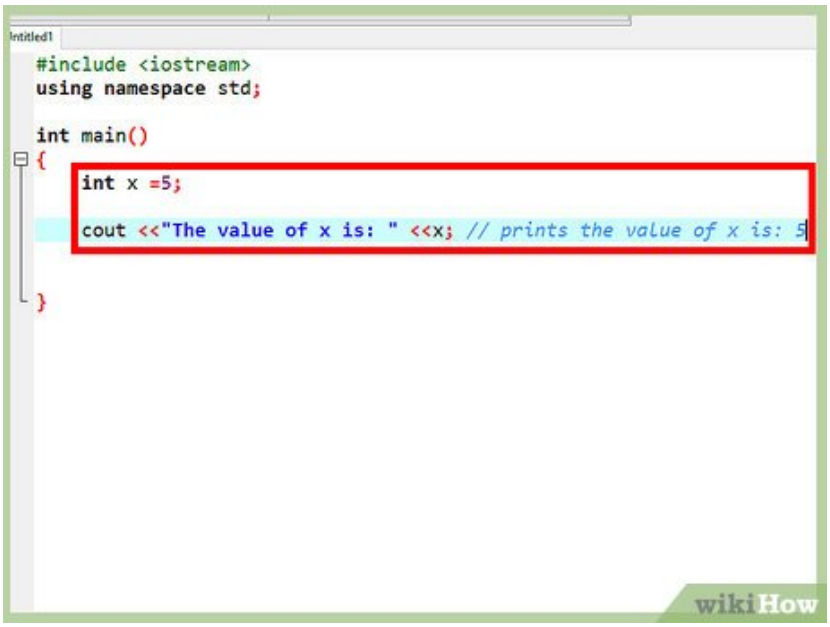
Write the cout statement. Within the main function, type the cout statement using the proper syntax. For example: cout 'type text here'; (or std::cout "type text here";, if you don't like the use of using directives)

3. 

Become familiar with other uses of cout. Cout can also be used to output the values of variables, as long the variable has already been defined. Simply write the name of the variable after the insertion operator as shown below.

Error: cout"x"; wouldn't print 5, it would print 'x' (as a char) The same goes for cout"y"; Also, cout doesn't implicitly add newlines, meaning the above example would print "xy", and if we fixed the bug to print the value of x and the value of y, it would print "523". The solution is to use a newline symbol. A newline symbol is written as n. Example: std::cout x "n"; would print the value of x, then a newline character, meaning if we print

the value of y (using the example above, it would print "5", then "23" on a new line.

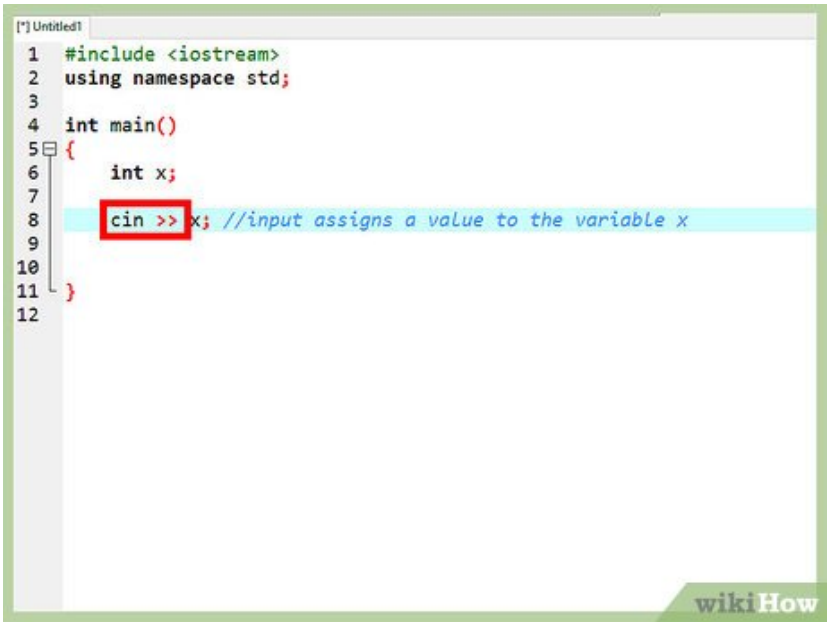
1. 

   Use multiple insertion operators in a single statement. Insertion operators can be simply chained together, one after the other as shown in the figure.

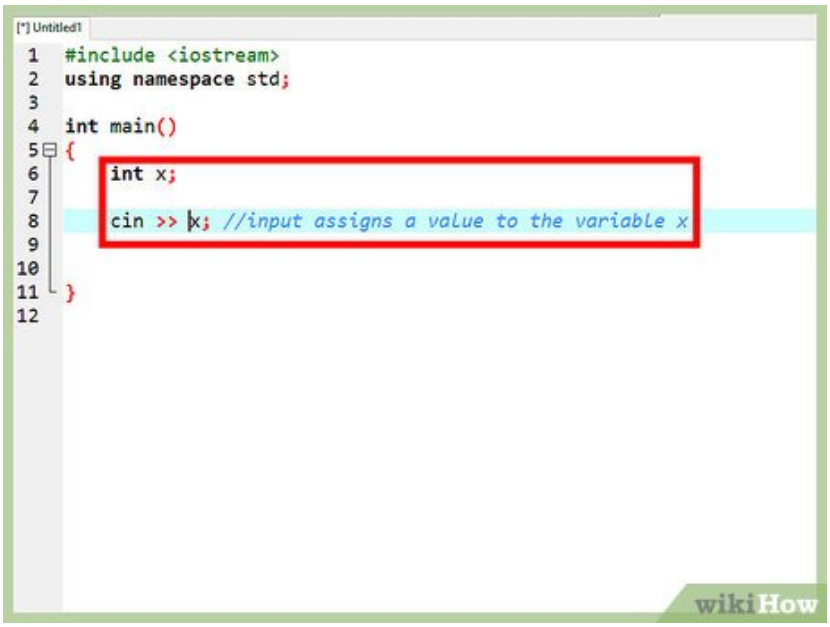
(Note, for advanced readers, ignore this otherwise: The reason why you can chain calls to cout is within the insertion operator (operator) itself. The insertion operator returns *this, which, in this context, is the first parameter (cout), meaning *this returns cout. Successive calls are then parsed as cout ..., which works.)

Part 3 of 3:

Writing Cin Statements

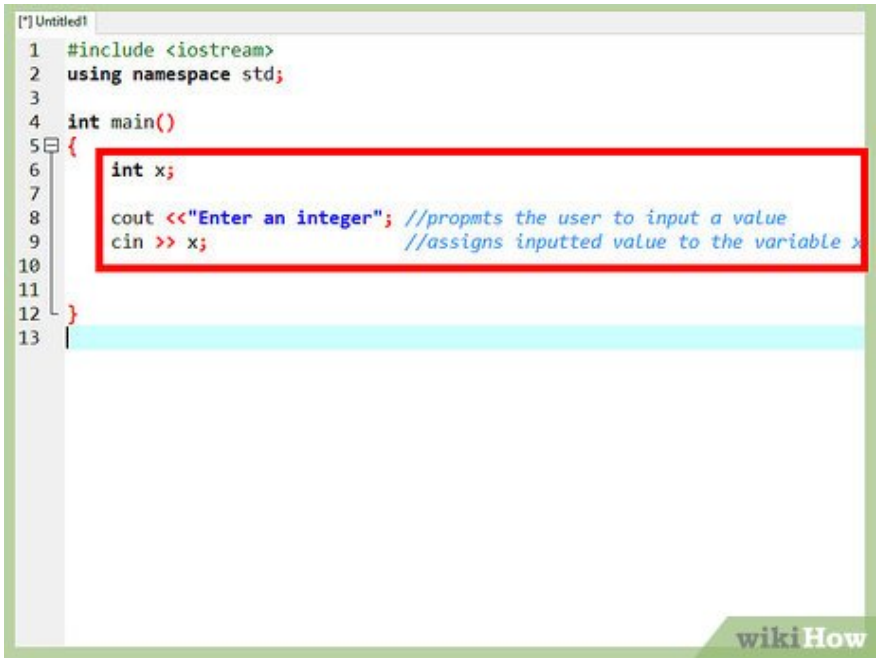
1. 

Know the syntax. Cin is used with the extraction operator, which is written as >> (two 'greater than' signs). The operator is then followed by a variable where the inputted data is stored. The line must end with a semicolon.

2. 

Write the cin statement. First declare a variable. Then write a cin statement to define a value for the variable as shown. When the program runs, the input that user enters will be assigned to the variable. Note that the cin statement does not output any text onto the monitor.

3. **Combine cin and cout statements.** Cin and cout statements can and should be used together. For instance, a cout statement may be used to prompt the user to assign a value to a variable, which is then assigned through a cin statement as shown in the figure.



You finished reading the article "**How to Use C++ to Write Cin and Cout Statements**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.
