

Hello, World

The first program that we usually write in any language we're learning is Hello, World. A Hello, World program just prints "Hello, World" to the screen. CodeBlocks is the development environment we're going to use for C++ in this class.

Step 1: Open CodeBlocks

Click on the blue and white icon in the upper left corner of the VM. A drop-down menu will appear with a textbox where you can type the name of the program you want to search for. In the textbox, type CodeBlocks. You will only need to type a portion of the word and the CodeBlocks program should show up in a list of programs that match your search. Click on the CodeBlocks program (**Figure 1**).

When CodeBlocks loads, you will be presented with the following screen (**Figure 2**) where you can create a new project, open an existing project, as well as a few other options. You can ignore this screen for now. In future classes, we will be creating projects, but not yet.

Figure 1: Searching for CodeBlocks

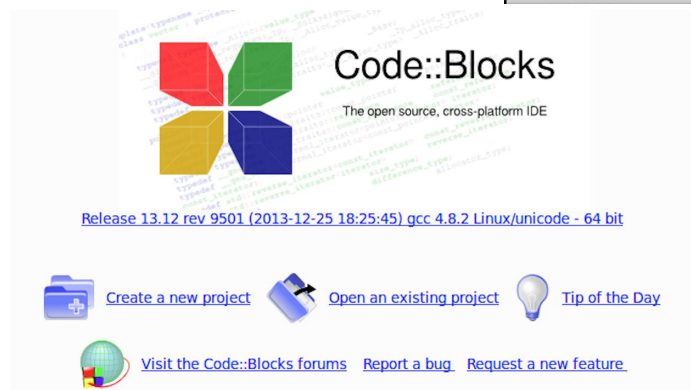
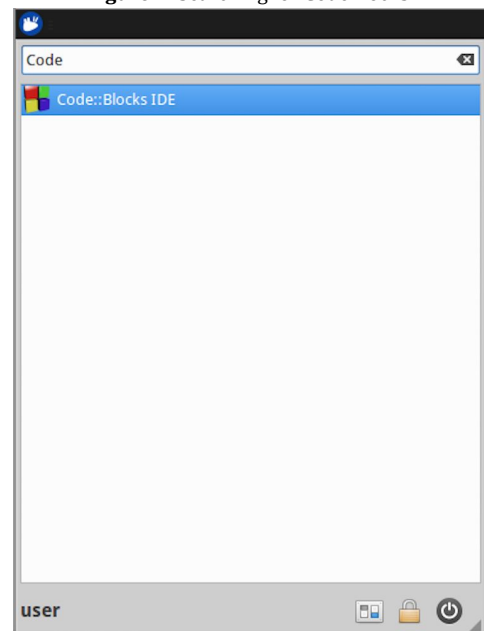


Figure 2:: CodeBlocks Welcome Screen

Step 2: Open an Empty File

Select File -> New -> Empty File. You can also click on the New File icon under the File menu. A new, blank file called Untitled1 will be opened. (**Figure 3**)

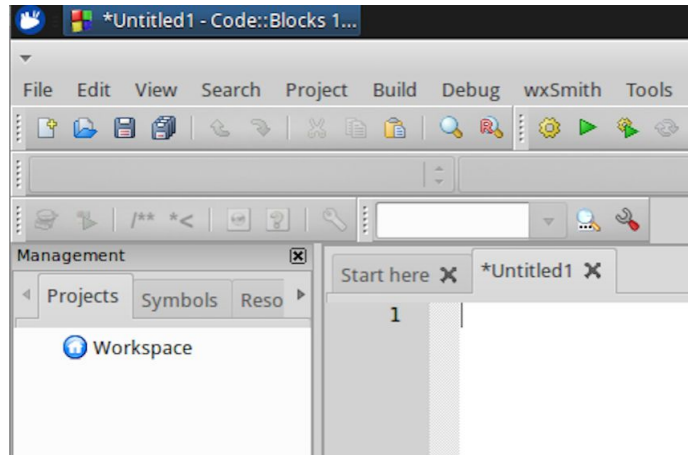


Figure 3: Empty File in CodeBlocks

Step 3: Your First Code!

Starting on line 1 in Untitled1, type the following code:

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6
7      cout<<"Hello, World";
8
9  }
10
```

Step 4: Saving Your File

Save the file. Name it **recitation2.cpp**. The .cpp extension on the filename tells CodeBlocks that the file should be read as C++ code. Once you save it, the lines in the file should be color coded to reflect what they do in the program. This is called *syntax highlighting* (**Figure 4**).

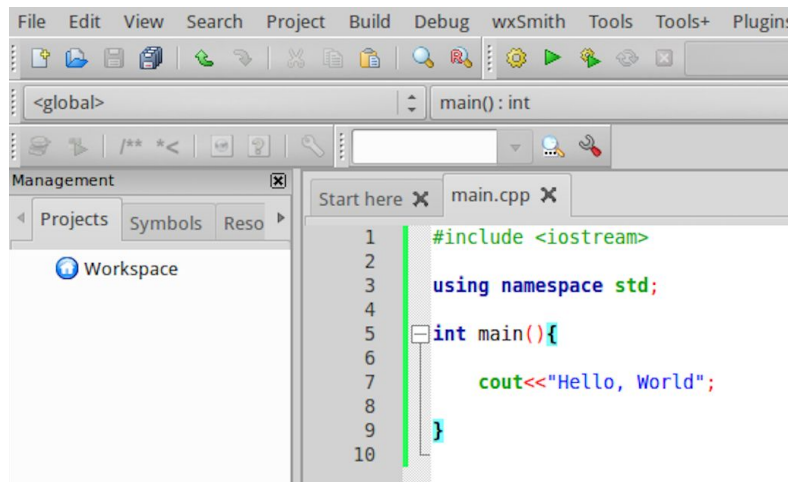
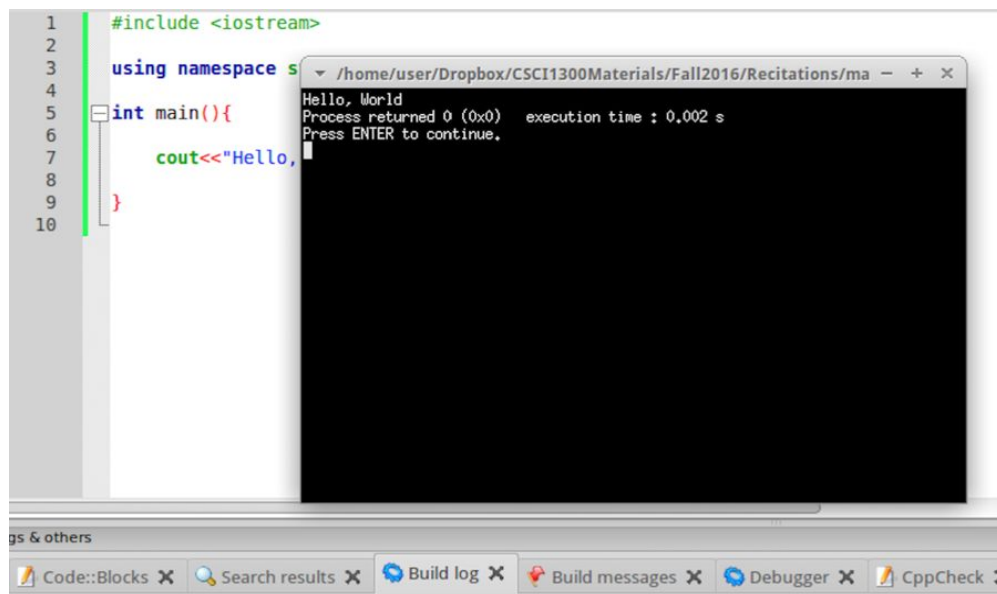


Figure 4: Syntax Highlighting

Important: You should save your work frequently in CodeBlocks to avoid losing your work in the event of the program crashing.

Step 5: Running Your Code

To run the program, click on the icon with the green arrow and yellow gears just under the Tools menu. If it works, you should see a terminal window load and print Hello, World.



Step 6: Update Your Code

Now, change the cout statement so that your program prints out the following:

Hello, CS1300 World!

Step 7: Submitting Your Code on COG

Throughout the semester we will be using the autograder, COG. The score that you receive on COG will automatically be sent to your Moodle gradebook and logged as your grade. **However, you may upload your assignment to COG as many times as you want to.** This enables you to maximize your score. To acclimate you to using COG, you will be uploading your recitation 2 assignment to COG.

Step A: .cpp File Name

First, you must ensure that your file is named **recitation2.cpp**.

Step B: Compress Your recitation2.cpp File to .zip

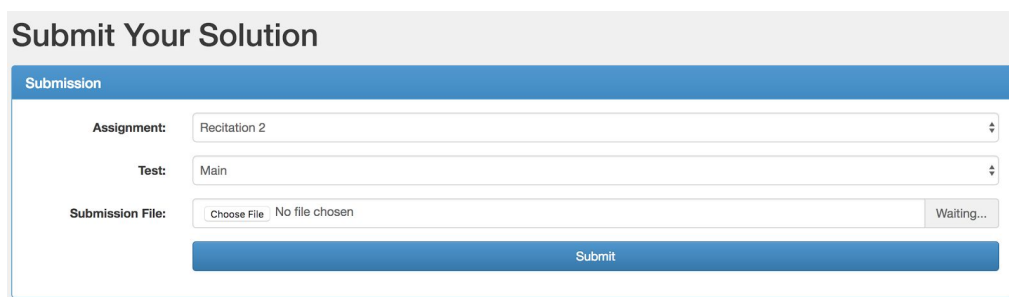
Second, you must compress your recitation2.cpp file into a .zip file. Locate your .cpp file, right click it, and chose the option to compress it.

Note: You *must* compress it as a .zip file - no other compressed formats are permitted.

Note: You can name you .zip file anything you want.

Step C: Upload .zip to COG Website

Next, navigate to <https://web-cog-csci1300.cs.colorado.edu>. You will then select 'Recitation 2' from the assignment drop-down menu. Next to 'Submission File', select the 'Choose File' option. This will open up a new window. Navigate to the location of your .zip file from step B and choose it for upload. Lastly, click the large blue 'Submit' button.



The screenshot shows a web form titled "Submit Your Solution". It has a blue header bar with the word "Submission" in white. Below the header, there are three rows of form fields. The first row is labeled "Assignment:" and has a dropdown menu with "Recitation 2" selected. The second row is labeled "Test:" and has a dropdown menu with "Main" selected. The third row is labeled "Submission File:" and has a button labeled "Choose File" and a text field showing "No file chosen". To the right of the text field is a button labeled "Waiting...". At the bottom of the form is a large blue button labeled "Submit".

Step D: Reading the COG Output

After you upload your .zip file and submit it, COG will compile your code and check its output against an expected output. This process will be output to the dialogue box that appears after pressing 'Submit.'

Important: If your code does not compile, you will receive a **ZERO**.

The dialogue box will give you insight into what parts of your code are incorrect and what the correct output should have been. For example, if you did not change the program from the original cout of "Hello, World", the program will take off 50% of the score and it will notify you of the expected cout.

Important: You must match the cout exactly as specified in the assignments for COG to award you credit.

Lastly, COG will tell you the score you received on the assignment. If you are not satisfied with this score, follow the advice in the COG output to debug your program. That is, rewrite it, compress it into a .zip file, and reupload it. You can do this as many times as you wish.

Step 8: Submitting Your Code to Moodle

In addition to submitting your code to COG, you **must** submit your files to Moodle. Utilize COG to finalize your assignment and maximize your score. Once you are satisfied with your grade, you must submit the files to Moodle.

Navigate to the correct week on Moodle and look for the following link under the Recitation section. Use this link to submit your **.cpp** file.



Recitation Coding Activity

During every recitation we will go over what you need to know to prepare for your weekly homework. This will be achieved by completing the recitation quiz on Moodle.

Notes on Best Coding Practices

The quality of a script lies in its implementation and in how clean and “readable” it is. There are several best practices that are worth to mention. For our first lab let’s focus on naming conventions. Please refer to the following [best practices](#) to solve the questions in CodeRunner.

Access the Recitation Programming Activity on Moodle

There are three problems for you to attempt and complete by the end of the recitation.

Note: *When you press the “check” at the bottom of the question, it will attempt to compile and run your version of the function with our test cases. The results will appear below your answer code.*

Some points to think about when you solve the questions:

- Do you have to return a value, to print it out to the console, or both?
- Are there parameters? How many?
- Make sure you use good naming conventions (see “***Notes on Best Coding Practices***”).

PA #1 - Modify the given code to implement a new functionality.

The first problem shows you a syntactically and semantically correct function declaration and code. Take inspiration from it to implement the code needed to achieve the new task required.

We need you to modify this code to change its behavior as listed in the question description.

PA #2 - You must write the code for the function declaration that we have provided. Inside the function definition implement the logic required to accomplish the task explained in the question.

PA #3 - You must derive the correct function declaration and write the code for the function code given a description. Write the complete function in the answer area. Once you are done with the activity raise your hand and show your work to the TA.