CSCI 1300 CS1:Starting Computing
Recitation 2
Instructor: Fleming/Gupta
Due: at the end of your recitation section

Objectives:

1. Write your first C++ program and submit it for autograding on Moodle
2. Other programming quiz questions in Moodle

Cloud 9 IDE, continued:

In Recitation 1, you created an account with Cloud9 and spun off your Private Workspace. One of the cool things about Cloud 9 is that every time you open your Private Workspace, it should look exactly like you left it when you closed your browser.

Open your Private Workspace and you should have the default files, two folders (rec1 and rec2), a few tabs at the bottom with the last commands we ran for unzipping files.

Running applications from the command line

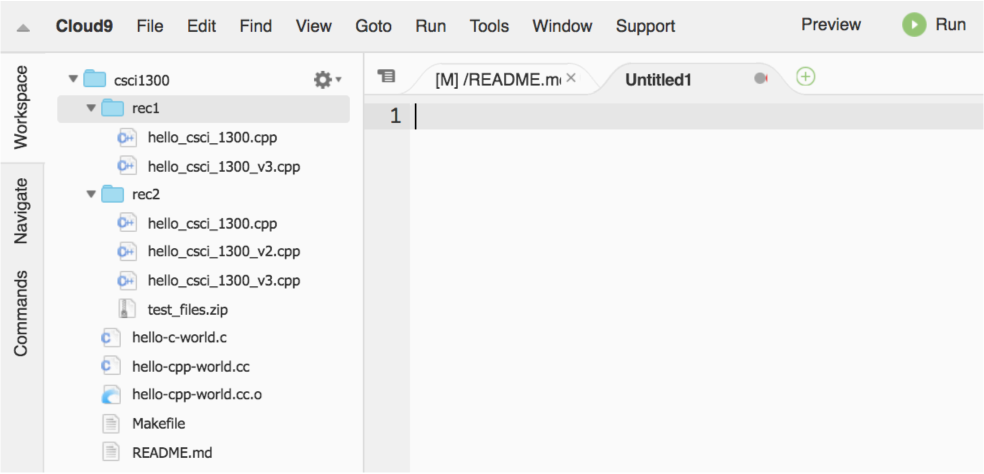
Many of the applications in Cloud9 can be run from the terminal. Running programs in the terminal is also called running them *from the command line*. In this recitation, you will run your first C++ program, run it from the command line, then copy the solution and submit it on Moodle for autograding.

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.

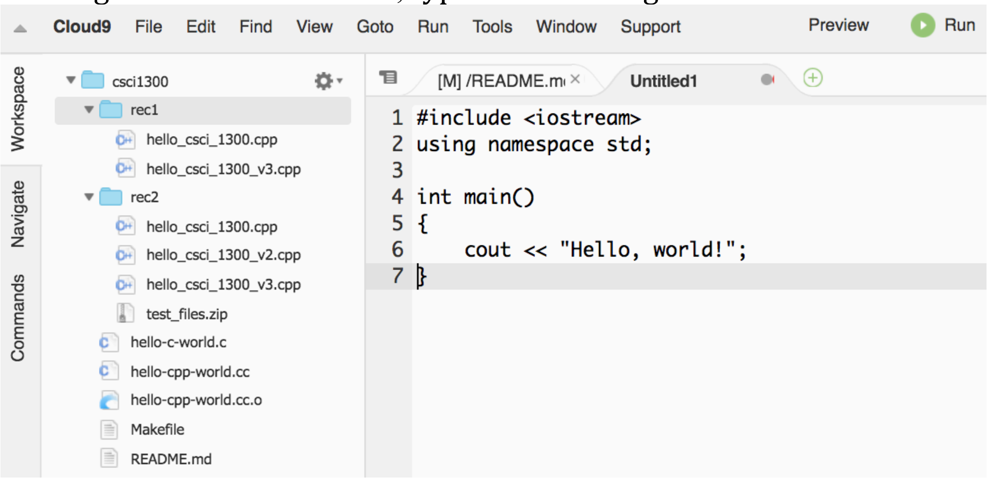
Step 1: Open an Empty File

In Cloud9, select File -> New -> New File. A new, blank file called Untitled1 will be opened.



Step 2: Your First Code!

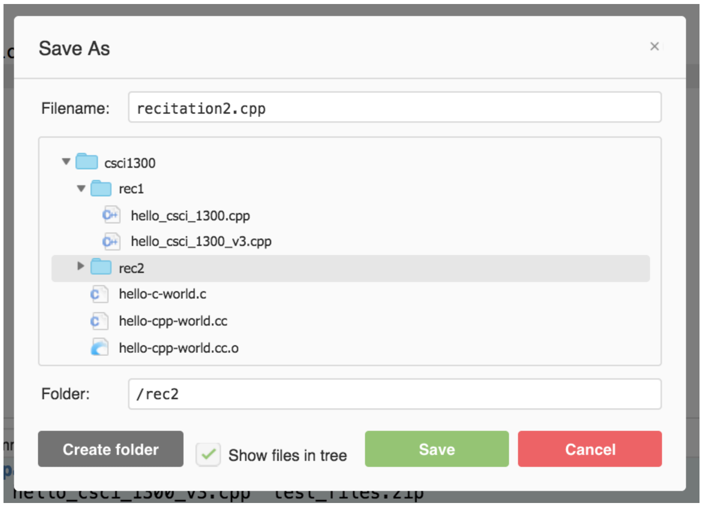
Starting on line 1 in Untitled1, type the following code:



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

int main()
{
    cout << "Hello, world!";
}
```

Step 3: Saving Your File

Save the file: go to File -> Save As... A dialog box will open. Name it **recitation2.cpp** and save it in the rec2 folder.

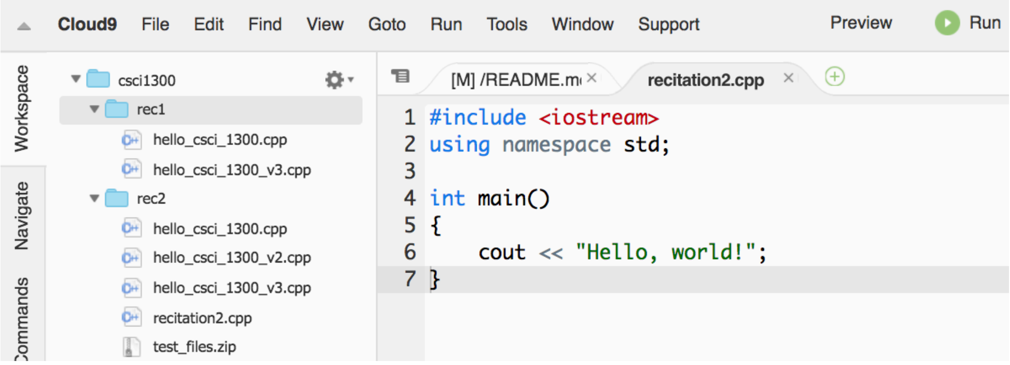


Filename: recitation2.cpp

Folder: /rec2

☑ Show files in tree

The .cpp extension on the filename tells Cloud9 that the file should be read in the C++ programming language. 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*.

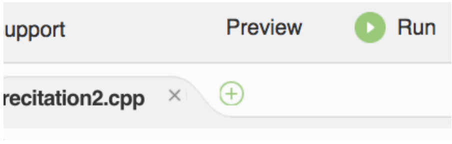


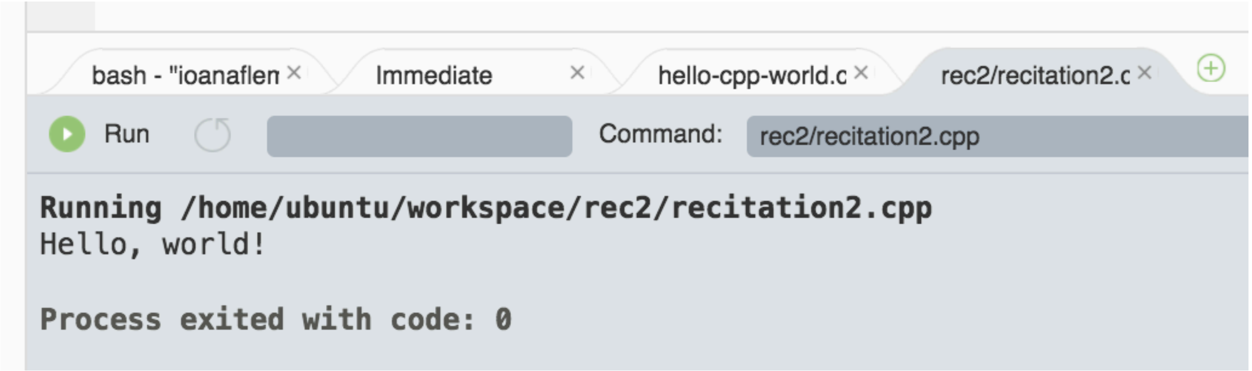
```
#include <iostream>
using namespace std;

int main()
{
    cout << "Hello, world!";
}
```

Important: You should save your work frequently in Cloud9 to avoid losing your work in the event of the program crashing.

Step 4: Running Your Code

To run the program, click on the icon with the green arrow next to the word Run. If it works, you should see new terminal tab window open at the bottom. The title of the tab shows the file being ran (rec2/recitation2.cpp), and inside the window you should see "Running ...." (again the name and full path of the file), and underneath it, the output of our program: Hello, world!





```
Running /home/ubuntu/workspace/rec2/recitation2.cpp
Hello, world!

Process exited with code: 0
```

Step 5: Running Your Code from Command Line

Move to the "bash" tab (the first tab in the bottom panel). Right-click again and Clear the Buffer. Make sure you are inside the *rec2* directory. Type:

```
$ g++ -g recitation2.cpp
```

the –g option turns on debugging, which we will use later in the semester, so we should get used to it.

This creates an executable called "a.out" (see figure below). You can run it by typing

```
$ ./a.out
```

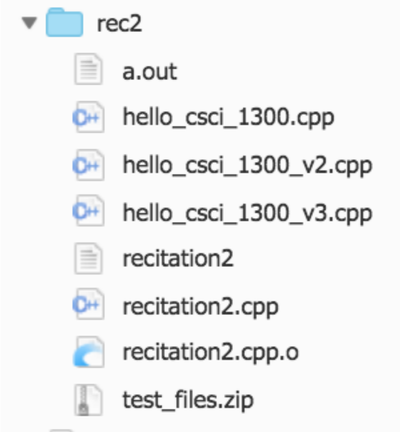
Since no executable name was specified to g++, a.out is chosen by default. You can alternatively use the "-o" option to change the name :

```
$ g++ -g -o recitation2 recitation2.cpp
```

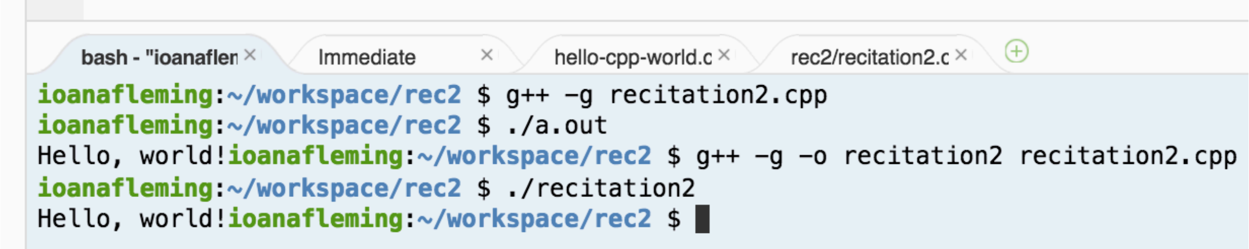
creates an executable called "recitation2" (see figure below). You can run it by typing

```
$ ./recitation2
```

Notice the output in the same: Hello, world!, followed by the return of the prompt, for new commands.



```
rec2
    a.out
    hello_csci_1300.cpp
    hello_csci_1300_v2.cpp
    hello_csci_1300_v3.cpp
    recitation2
    recitation2.cpp
    recitation2.cpp.o
    test_files.zip
```



```
ioanafleming:~/workspace/rec2 $ g++ -g recitation2.cpp
ioanafleming:~/workspace/rec2 $ ./a.out
Hello, world!ioanafleming:~/workspace/rec2 $ g++ -g -o recitation2 recitation2.cpp
ioanafleming:~/workspace/rec2 $ ./recitation2
Hello, world!ioanafleming:~/workspace/rec2 $ 
```

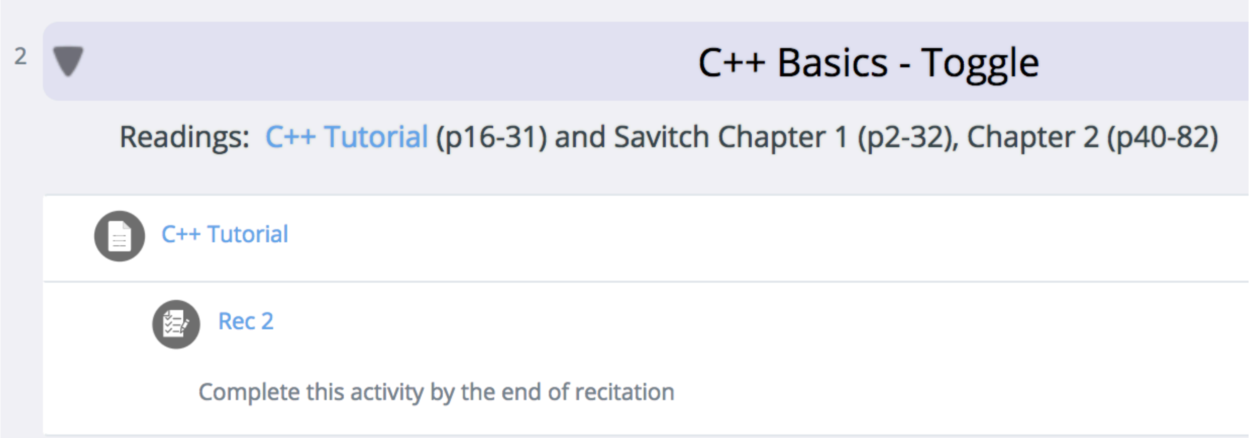
Step 6: Update Your Code

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

Hello, CS1300 World!

Step 7: Submitting Your Code on Moodle

Throughout the semester we will be using the Moodle autograder. Take a look at the first question from the *Rec2* activity on Moodle:



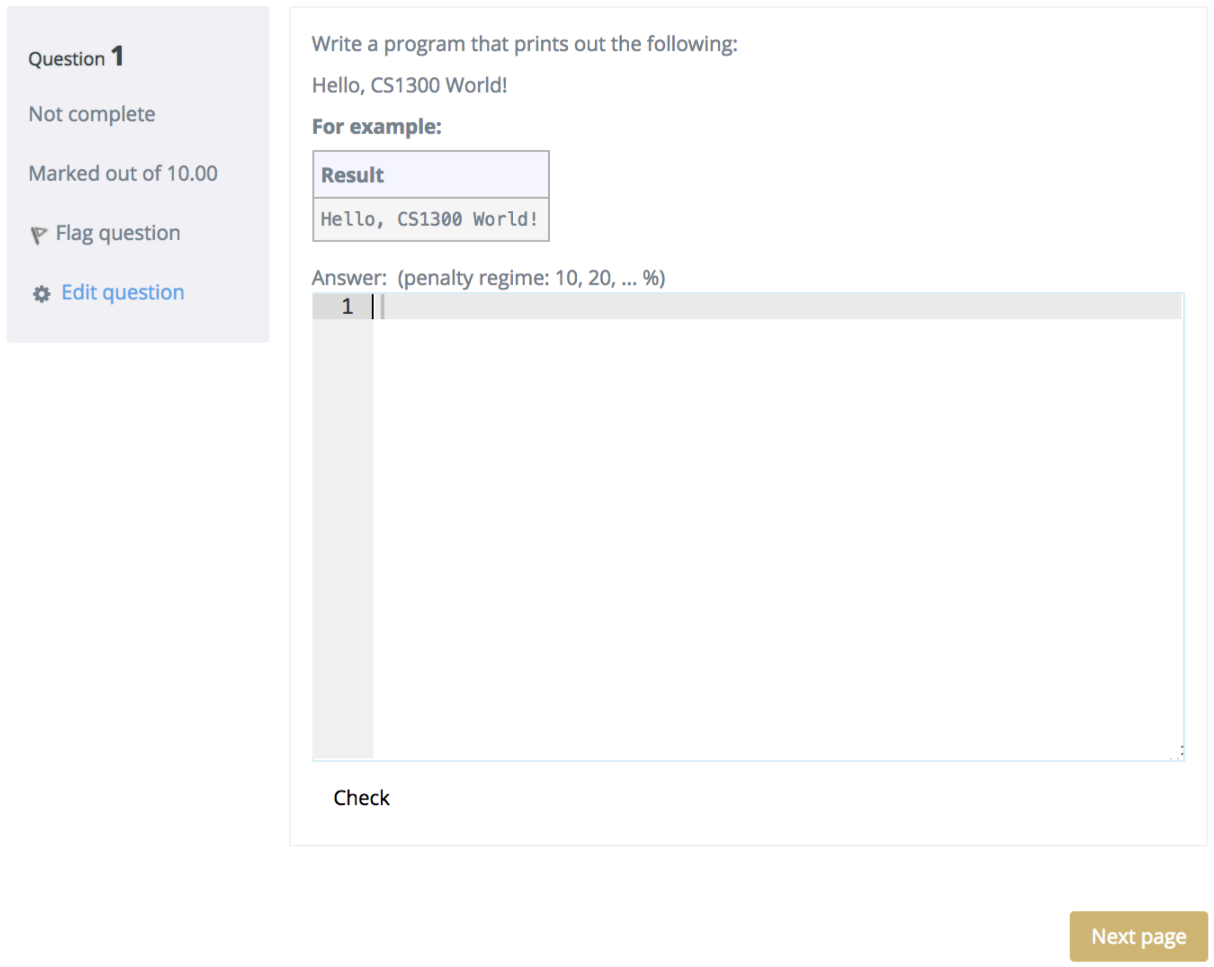
2 C++ Basics - Toggle

Readings: C++ Tutorial (p16-31) and Savitch Chapter 1 (p2-32), Chapter 2 (p40-82)

C++ Tutorial

Rec 2

Complete this activity by the end of recitation



Write a program that prints out the following:

Hello, CS1300 World!

For example:

Result
`Hello, CS1300 World!`

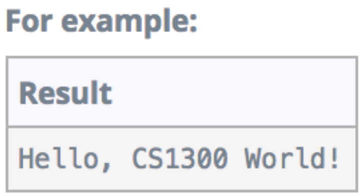
Answer: (penalty regime: 10, 20, ... %)

You can see:

1. The description of the problem:

Write a program that prints out the following:
Hello, CS1300 World!

2. An example of the program's output:



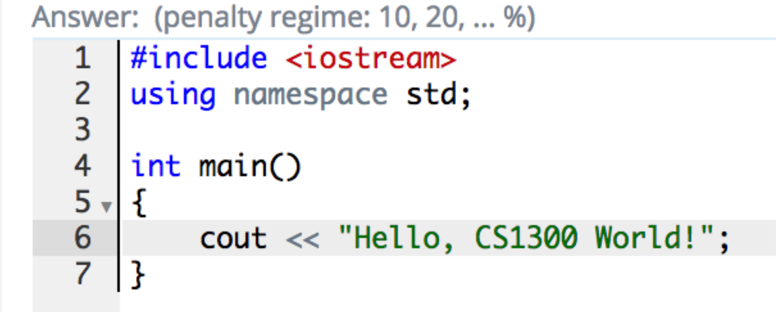
For example:

Result
`Hello, CS1300 World!`

3. The Answer box – this is where you will paste your code
4. The "Check" button – you can click this button as many time as you wish to check if the solution/code you entered in the Answer box is correct
5. On the left side you can see the problem is worth 10 points, and
6. At the bottom right there is a button "Next Page" which will lead you to the next problem

What you need to do:

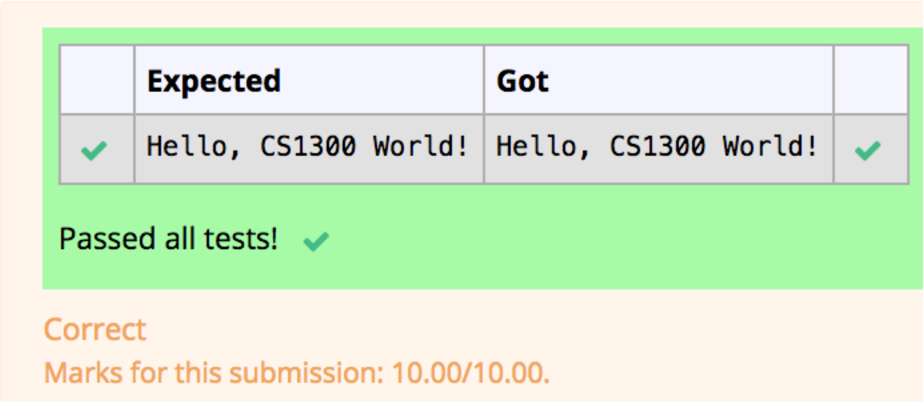
1. Copy your code from Cloud9 from the updated file *recitation2.cpp*
2. Paste it into the Answer box:



Answer: (penalty regime: 10, 20, ... %)

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

int main()
{
    cout << "Hello, CS1300 World!";
}
```

3. Press the Check button. If you code is correct, and the output from your solution matches the expected output, you should see green check marks indicating you have passed the test:



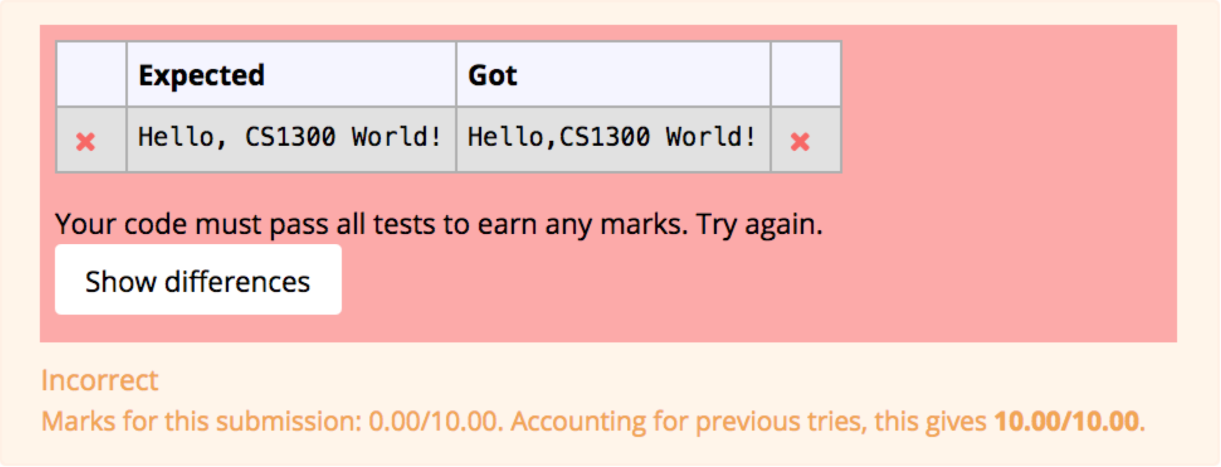
	Expected	Got	
✓	`Hello, CS1300 World!`	`Hello, CS1300 World!`	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

4. In the event you have a typo in the output message, the test will show the Expected output and the output from your solution (in the Got column). Look at the differences; update your solution to match the Expected output. The button Show Differences should help identify what is missing (or extra). In the example below, the solution forgot to include a “space” after the comma:

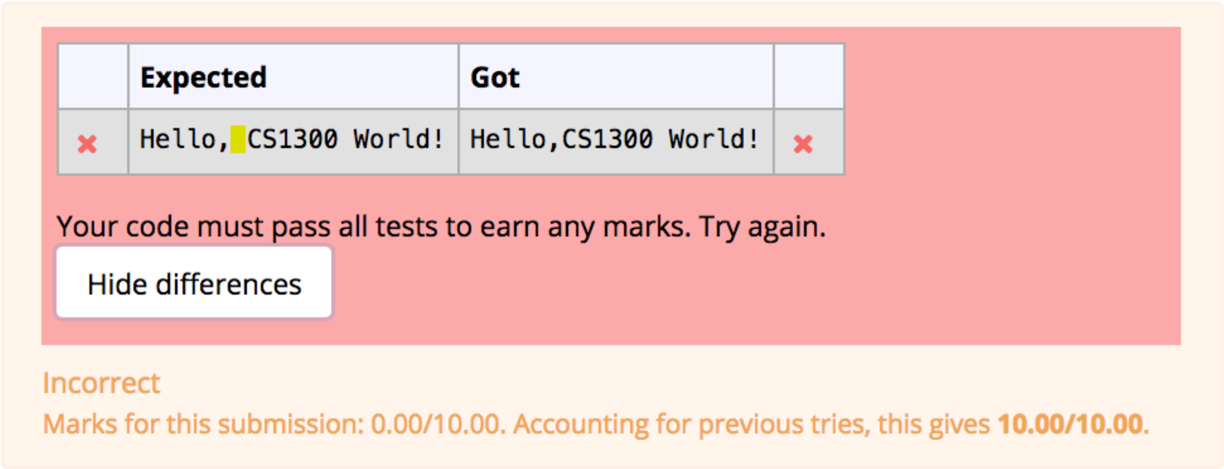


	Expected	Got	
✗	`Hello, CS1300 World!`	`Hello,CS1300 World!`	✗

Your code must pass all tests to earn any marks. Try again.

Incorrect

Marks for this submission: 0.00/10.00. Accounting for previous tries, this gives **10.00/10.00**.



	Expected	Got	
✗	Hello, CS1300 World!	Hello,CS1300 World!	✗

Your code must pass all tests to earn any marks. Try again.

Incorrect

Marks for this submission: 0.00/10.00. Accounting for previous tries, this gives **10.00/10.00**.

Note: notice that since the first attempt was correct, the problem is still scored at 10 points, “Accounting for previous tries, this gives 10.00/10.00”

Good Practice: <u>back up your work!</u>

It is recommended you use a backup system for your work. Get a Dropbox account or use Google Drive. Dropbox accounts are limited, while a CU Google Drive allows up to 1 TB (that should be enough for all your work, for all the courses you will take at CU).

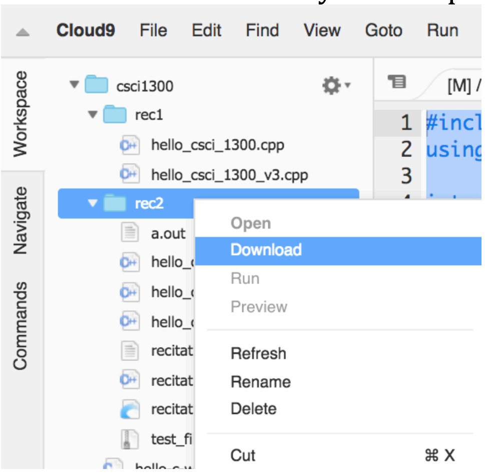
In order to backup our work, we first need to bring it out from Cloud9. It’s very simple, and you have multiple options:

1. Download just one file (the solution file you just submitted).

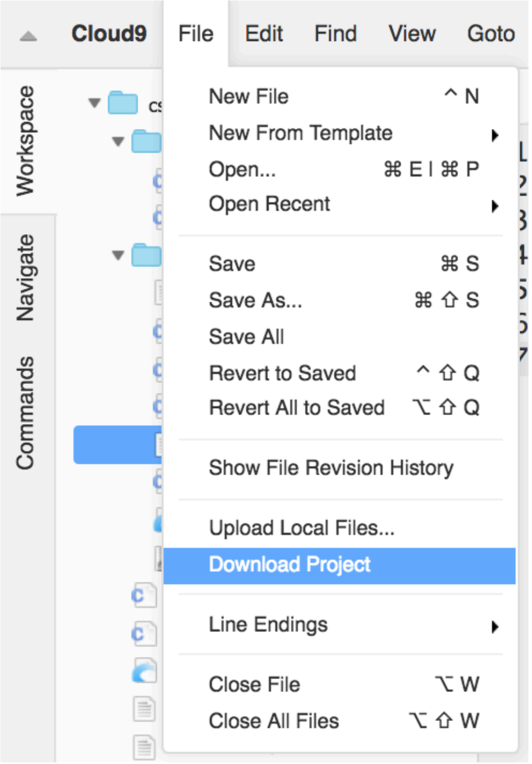
   Right click on the file name in the left panel (the file tree) and choose the option Download. Only one file will be downloaded, on your computer, at your default download location (your Downloads or My Downloads folder)

2. Download the entire *rec2* folder.

   Right click on the folder’s name and choose the option Download. An archive .zip file will be downloaded on your computer, with all the files from the *rec2* folder.



3. At any time, you can download the entire Cloud9 workspace. This means all the files you have created in Cloud9 since you first logged in. Go to File -> Download Project . This will download an archive .zip file, containing a folder named "Workspace", which includes all the subfolders you created, with all the files.



Other Moodle Programming Questions

Problems 2, 3, and 4 are also programming questions, but you don't have to paste in the Answer box the entire program you wrote. We refer to them as "Create a function" questions. In these 3 problems for **Recitation 2**, you will notice that some of the code was already written for you, as a starting point, in the Answer box. You will need to modify/update the code given to obtain the Expected Output.

Remember: you can press Check as many times as you want. It will show you what the Expected Output is, and also what is the output produced by your code. Compare the two outputs and make adjustments to your code. Use the Show Differences button to help you figure out what's missing, or what you might have extra. Good luck!

PA #2 - Modify the given code to implement a new functionality.
The second 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. Notice that part of the Expected Output is already accounted for in the provided code. Modify the given code to make it output a second line of text.

PA #3 - You must write the code inside the function body. Compute the time it takes for the sun to traveled to a given distance *dist*. Then, separate the time into the number of minutes and the number of seconds. Remember, we need variables to hold the values we compute.

PA #4 – For the last problem, you must write the entire function. Take inspiration from the previous two problems to write the correct function header. Inside the function's body, you need to make one computation, and then return the computed value.

Once you are done with the activity raise your hand and show your work to the TA.