CSCI 1300 CS1: Starting Computing
Ashraf, Cox, Spring 2020
Homework 2
Due: Saturday, February 1, by 6 pm
(5 % bonus on the total score if submitted by 11:59 pm January 31)

Objectives

- Compile and run C++ code
- Take user inputs and produce outputs
- Understand C++ data types
- Perform arithmetic operations

You can find the summary note (hw2 note: HelloWorld, data type) on Moodle.

Submissions

- **Conceptual reviews(mcq)**. There are a few multiple-choice questions to check your conceptual understanding. Don't forget to complete them!
- **C++ files**. All files should be named as specified in each question, and they should compile and run on Cloud 9 to earn full points. TAs will be grading styles of your code and comments. Please see the style guide on Moodle and the summary note on Moodle. At the top of each file, write your name with the following format:

```
// CS1300 Spring 2020
// Author: Punith Sandhu
// Recitation: 123 – Favorite TA
// Homework 2 - Problem # …
```

- **Code runner**. Your program will be graded by Code Runner. You can modify your code and re-submit (press "Check" again) as many times as you need to, until the assignment due date.

Plan for this week

Questions 1 - 3 are considered as recitation questions. You should be able to finish at least 3 questions during the recitation time. If you're able to finish them earlier, then continue working on the other questions and complete the homework before the deadline.

Questions

Question 1 (5pt): Hello World

The first program that we usually write in any language we're learning is *Hello, World*. Create a program that prints "`Hello, World!`" to the screen (the console window in Cloud9).

Expected output

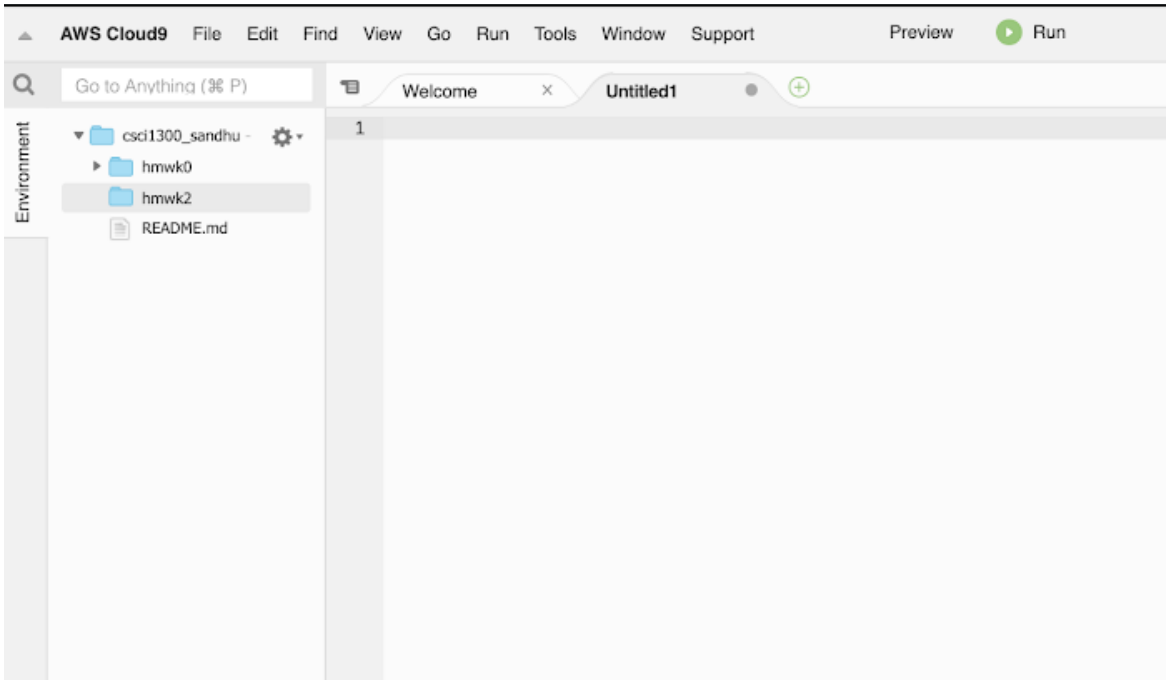
```
Hello, World!
```

The file should be named `helloWorld.cpp`

Here are some suggested steps:

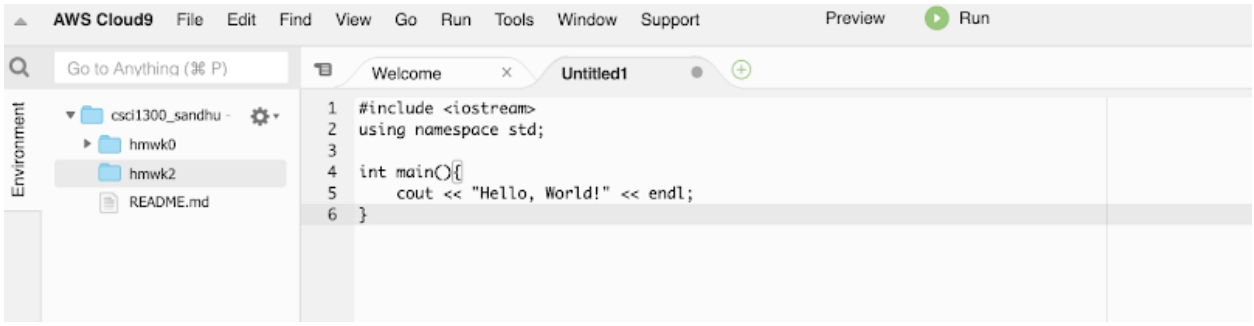
Step 1: Open an Empty File

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



Step 2: Your First Code

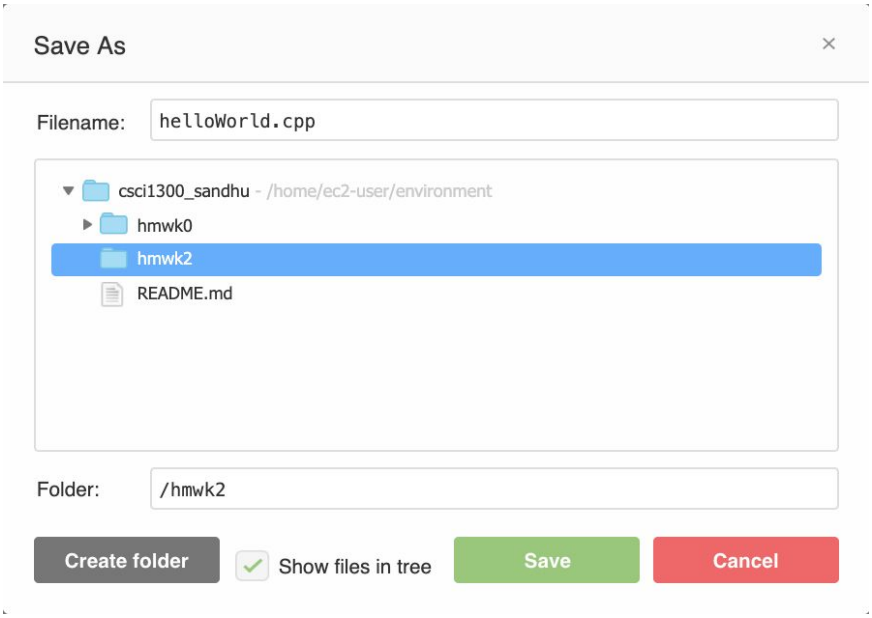
Starting on line 1 in Untitled1, type the following code.



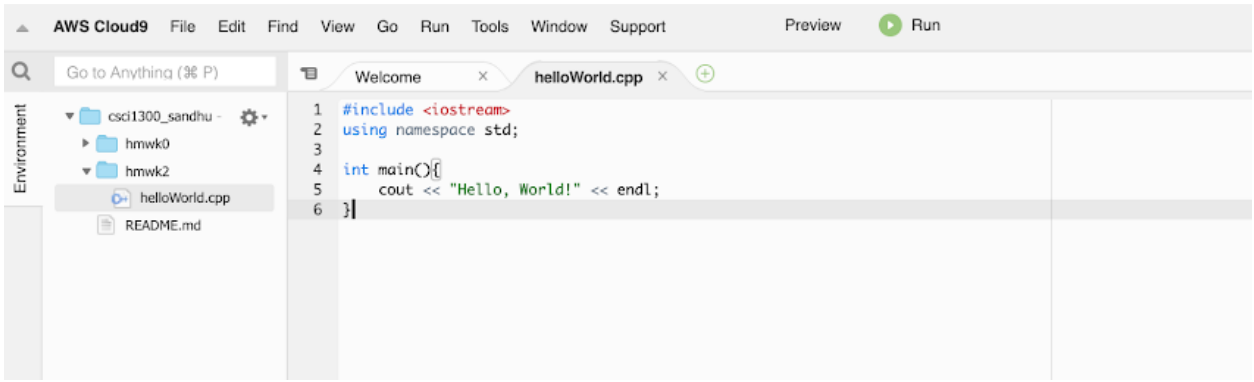
Step 3: Saving Your File

Save the file: go to File -> Save As... A dialog box will open. Name it **helloWorld.cpp** and save it in the **hmwk2** folder.

Note: *make sure you save it with the .cpp extension or it will not compile correctly!*

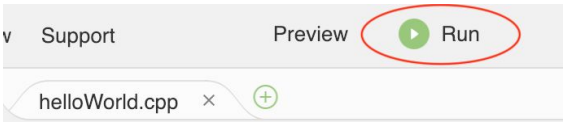


The `.cpp` extension on the filename tells Cloud9 that the file should be read in the C++ programming language. Once you save it, the lines in the file should be color-coded to reflect what they do in the program. This is called syntax highlighting.



Important: You should save your work frequently in Cloud9 to avoid losing your work in the event of the program crashing. Also, you should take a backup of your Cloud 9 environment.

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 run (hmwk2/ helloWorld.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!`"

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 **hmwk2** directory. Type:

```
$ g++ helloWorld.cpp -g -std=c++11
```

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

the `-std=c++11` option makes sure that the c++ version used to run the program is c++ 11. If you don't give this option then default version(which is usually C++98) is used.

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++ helloWorld.cpp -g -std=c++11 -o hello
```

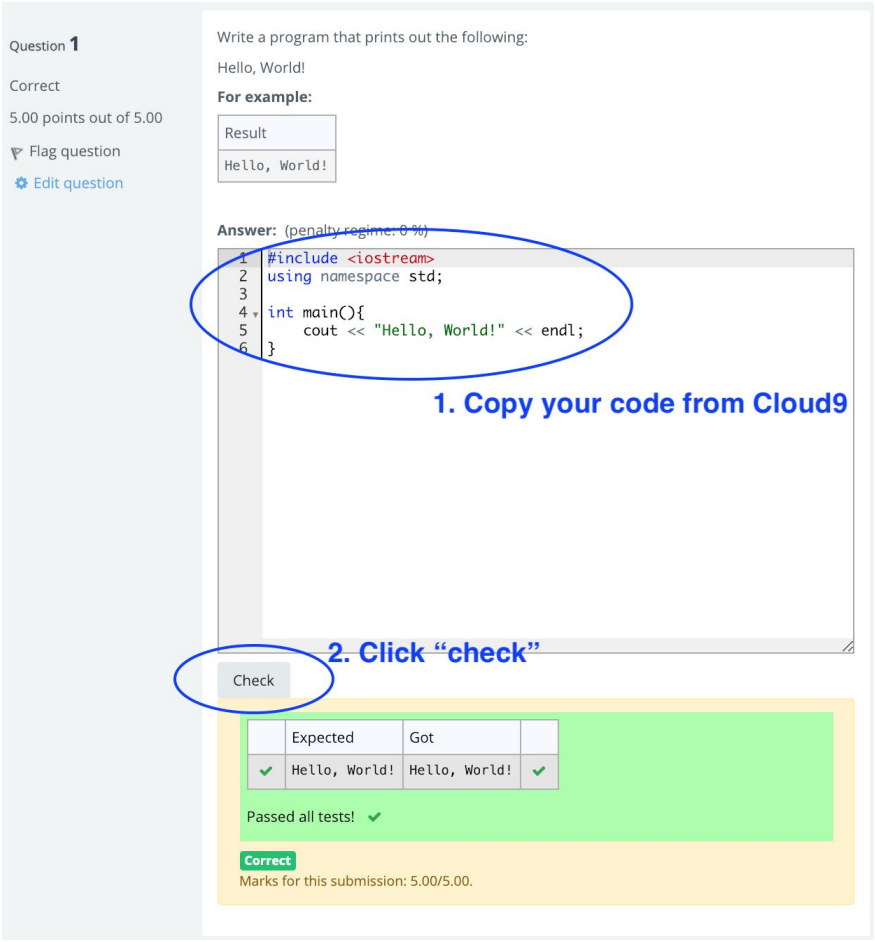
creates an executable called "`hello`" (see figure below). You can run it by typing

```
$ ./hello
```

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

Step 6: Submit to Moodle CodeRunner

Head over to Moodle to the link **Homework 2 CodeRunner**. Submit your solution for the first problem and press the Check button. You will see a report on how your solution passed the tests, and the resulting score for the first problem. You can modify your code and re-submit (press “Check” again) as many times as you need to.



If the code runner says “incorrect”, click “show difference” button. It will highlight the differences between the expected output and the output your program produced. These outputs should be exactly the same, including upper/lower case, punctuation, and spaces. Otherwise, it’ll mark as incorrect. It’s a computer. It needs to be precise.

Question 2 (5pt): Hello You!

If a program is more interactive, it's fun! Create a program that takes a name and print prints "`Hello, <name>!`". Your output should be exactly the same as below, including the prompt for the user input.

Expected output (**bold** is user input)

```
Enter your name:
Malvika
Hello, Malvika!
```

The file should be named as `helloYou.cpp`. Don't forget to head over to CodeRunner on Moodle and paste your solution in the answer box!

Question 3 (10pt): Carnot efficiency

In thermodynamics, the Carnot efficiency is the maximum possible efficiency of a heat engine operating between two reservoirs at different temperatures. The Carnot efficiency is given as

$$\eta = 1 - \frac{T_C}{T_H}$$

where T_C and T_H are the absolute temperatures at the cold and hot reservoirs, respectively. Write a program that takes temperatures at the cold receiver(T_c) and hot receiver(T_h) as `integer` values, and computes η the Carnot efficiency.

Expected output (**bold** is user input)

```
Enter cold reservoir temperature:
10
Enter hot reservoir temperature:
80
Carnot efficiency: 0.875
```

The file should be named as `carnot.cpp`. Don't forget to head over to Code Runner on Moodle and paste your solution in the answer box!

Question 4 (10pt): Calculating sphere volume and area

Create a program that takes a radius as a floating-point value (as `double`), and it prints both the volume and surface area of a sphere with the given radius. The formulas are:

$$volume = \frac{4}{3}\pi r^3$$

$$surfaceArea = 4\pi r^2$$

For π, use `M_PI` from `#include <cmath>`

Expected output (**bold** is user input)

```
Enter a radius:
3.3
Volume: 150.533
Surface area: 136.848
```

The file should be named as `sphereVolumeArea.cpp`. Don't forget to head over to Code Runner on Moodle and paste your solution in the answer box!

Question 5 (10pt): Temperature conversion

In science, the temperature is always described in Celsius, but in the U.S. we tend to use Fahrenheit. Create a program that takes user input for Fahrenheit (as `double`) and converts it into Celsius. The formula is:

$$Celsius = \frac{5}{9}(Fahrenheit - 32)$$

The calculated temperature Celsius values should be formatted with a two-digit precision as shown below. Fahrenheit value should be printed without any precision formatting applied.

Expected output (**bold** is user input)

```
Enter a temperature in Fahrenheit:
76
The temperature 76 degrees Fahrenheit is 24.44 degrees Celsius.
```

The file should be named as `temperature.cpp`. Don't forget to head over to Code Runner on Moodle and paste your solution in the answer box!

Question 6 (10pt): Convert seconds

Write a program that takes a number of seconds as user input (as an `integer`) and converts it to hours, minutes, and seconds as shown below. You should convert the amount of time in such a way that maximizes the whole numbers of hours and minutes.

Expected output 1 (**bold** is user input)

```
Enter a number of seconds:
60
0 hour(s) 1 minute(s) 0 second(s)
```

Expected output 2 (**bold** is user input)

```
Enter a number of seconds:
3671
1 hour(s) 1 minute(s) 11 second(s)
```

The file should be named as `convertSeconds.cpp`. Don't forget to head over to Code Runner on Moodle and paste your solution in the answer box!

Question 7 (10pt): Population

The U.S. Census provides information about the current U.S. population as well as approximate rates of change. Using those rates and the current US population, write a program that takes a current population and computes the U.S. population in exactly one year (365 days). If you end up with a non-integer projected population, then round down to the nearest whole person.

Three rates of change are provided:

- There is a birth every 8 seconds
- There is a death every 12 seconds
- There is a new immigrant arriving in the US every 27 seconds

Expected output (**bold** is user input)

```
Enter the current population:
1000000
The population in one year: 3482000
```

The file should be named as `population.cpp`. Don't forget to head over to Code Runner on Moodle and paste your solution in the answer box!

Homework 2 checklist

Here is a checklist for submitting the assignment:

1. Complete the **conceptual reviews(mcq)**
2. Complete the code **Homework 2 CodeRunner**
3. Submit one zip file to **Homework 2**. The zip file should be named, **hmwk2_lastname.zip**. It should have the following 7 files:
    - **helloWorld.cpp**
    - **helloYou.cpp**
    - **carnot.cpp**

- **sphereVolumeArea.cpp**
- **temperature.cpp**
- **convertSeconds.cpp**
- **population.cpp**

Homework 2 points summary

Criteria	**Pts**
Conceptual reviews (MCQ)	10
CodeRunner (problem 1 - 7)	60
C++ file submission (compiles and runs, style and comments)	30
Recitation attendance (Week 3)*	-30
Total	100
5% early submission bonus	+5%

* if your attendance is not recorded, you will lose points. Make sure your attendance is recorded on Moodle.