CSci 1113: Introduction to C/C++ Programming for Scientists and Engineers Homework 2 Spring 2020

Due Date: Friday, February 21, 2020 before 11:55pm.

Instructions: This is an individual homework assignment. There are two problem worth 20 points each. Solve the problem below by yourself (unlike the labs, where you work collaboratively), and submit the solution as a C++ source code file. Here are a few more important details:

1. Unlike the computer lab exercises, this **is not** a collaborative assignment. You must design, implement, and test the solution to each problem on your own without the assistance of anyone other than the course instructor or TAs. In addition, you may not include solutions or portions of solutions obtained from any source other than those provided in class: examples from the textbook, lectures, or code you and your partner write to solve lab problems. Otherwise obtaining or providing solutions to any homework problems for this class is considered academic misconduct. See the "collaboration rules" file on the class website page for more details, and ask the instructor if you have questions.

2. Because all homework assignments are submitted and tested electronically, the following are important:

- You follow the naming conventions mentioned at the end of the problems.
- You submit the correct file(s) on gradescope (https://www.gradescope.com/) by the due deadline.
- You follow the example input and output formats exactly given in each problem description.
- **Regardless of how or where you develop your solutions, your programs compile and execute on gradescope computers (which run Linux/Ubuntu operating system like the cselabs machines).**

3. The problem descriptions will usually show at least one test case and the resulting correct output. However, you should test your program on other test cases (that you make up) as well. Making up good test cases is a valuable programming skill, and is part of ensuring your code solution is correct.

Problem A: Number guessing game (20 points)
Write a C++ program to do the following number guessing game. First, the computer should ask you to enter a "seed" value (see below). Then pick a random number from 1 to 9 without telling you the number. You get three chances to try and guess the number the computer picked. After you make one of your guesses, the computer should tell you one of the following statements:

1. "Close!" if you are within 1 of correct answer.
2. "Too low!" if you are below the number and not within 1.
3. "Too high!" if you are above the number and not within 1.
4. "Correct!" if you guessed the number exactly.

If the player has not guessed the correct number after 3 tries, inform the player that they lost the game, tell them the correct answer and end the program. If the user correctly guesses the number ("Correct!"

above), end the program.

Important: Make sure your first lines of code in main() are as shown below to ask for the seed and setup the random number (i.e. variable “random”):

```
     int seed;
     cout << "Enter seed:\n";
     cin >> seed;
     srand(seed);
     int random = rand()%9+1;
```

If you do not follow this process, or generate a random number in a different way, you will probably not pass the test cases. You should also add this include at the top:

```
     #include <cstdlib>
```

Example 1 (user input is underlined):

```
Enter seed:
5
What is your guess?
1
Too low!
What is your guess?
5
Too low!
What is your guess?
9
Close!
You lost, the number was 8
```

Example 2 (user input is underlined):

```
Enter seed:
5
What is your guess?
9
Close!
What is your guess?
8
Correct!
```

Example 3 (user input is underlined):

```
Enter seed:
2
What is your guess?
9
Too high!
What is your guess?
5
Too low!
What is your guess?
999999999
Too high!
You lost, the number was 7
```

When you are done, name the source code file hw2A.cpp. Then log into gradescope and upload your file for the “Homework 2A” submission. **If you name your file incorrectly it will be unable to compile and run your code, so you will fail all test cases**. You may submit cpp files as many times as you want until the deadline to try and fix the code if you fail a test case. Following rigorous naming conventions and using test cases are something computer programmers often must do in “real life” programming, and so submitting your program with the correct name and functionality is part of doing this assignment correctly.

Problem B: Big (bad) integer calculator (20 points)
Write a calculator that ignores order of operations and just goes left-to-right. The format will always be: [integer] [operator] [integer] [operator] ... [integer] [# symbol]. You can assume that they will enter in this format and you should be able to output an answer (your program should handle any type of spacing, such as “1+2#” or “1+   2 #”). The only operators you need to handle are: +, -, / and *.

This should still do integer division, so if they enter “1+4/6#” it would then compute the addition (no order of operations) and get “5/6” then do integer division and get “0” as the final output (example 1).

Example 1 (user input underlined):
```
Enter a formula:
1+4/6#
0
```

Example 2 (user input underlined):
```
Enter a formula:
2+3*2/5000+9-4#
5
```

Example 3 (user input underlined):
```
Enter a formula:
2#
2
```

When you are done, name the source code file hw2B.cpp. Then log into gradescope and upload your file for the “Homework 2B” submission. **If you name your file incorrectly it will be unable to compile and run your code, so you will fail all test cases**. You may submit cpp files as many times as you want until the deadline to try and fix the code if you fail a test case. Following rigorous naming conventions and using test cases are something computer programmers often must do in “real life” programming, and so submitting your program with the correct name and functionality is part of doing this assignment correctly.

Problem C: Big (bad) integer calculator v2.0 (5 points extra credit)
In part B you made a calculator that detected the pattern: [integer] [operator] [integer] [operator] ... [integer] [# symbol]. A case we **do not** expect you to handle for part B is if someone just entered a single ‘#’. For this extra credit problem, handle this input, along with all the other normal computation. If the user enters just ‘#’, display “Error”.

Hint: you might have to look up how to convert between different types.

Example 1 (user input underlined):
```
Enter a formula:
#
Error
```

Example 2 (user input underlined):
```
Enter a formula:
2+3*2/5000+9-4#
5
```

Example 3 (user input underlined):
```
Enter a formula:
42#
42
```

When you are done, name the source code file hw2C.cpp. Then log into gradescope and upload your file for the “Homework 2C” submission. **If you name your file incorrectly it will be unable to compile and run your code, so you will fail all test cases**. You may submit cpp files as many times as you want until the deadline to try and fix the code if you fail a test case. Following rigorous naming conventions and using test cases are something computer programmers often must do in “real life” programming, and so submitting your program with the correct name and functionality is part of doing this assignment correctly.