Lab 5

In order to get credit for the lab, you need to be checked off by the end of lab. You can earn a maximum of 3 points for lab work completed outside of lab time, but you must finish the lab before the next lab. For extenuating circumstance, contact your lab TAs and Instructor.

This part of the lab cannot be completed in pairs!!!

(7 pts) 50 minute Practice Proficiency Demo

1. First, open a terminal on the host computer if necessary:
   - Once setup, right click "Open Terminal"
   - Make sure you type vim to use vim, not just vi, and create test.cpp
   - Nano, vi, or emacs are also acceptable text editors.
2. Get 2 questions from the TA. Choose one question you want to solve. All questions include variables, user input, conditional execution, and repetition. You should be able to finish the code within one hour.
   - Scoring:
   - You get max 7 points for fully coding the solution
   - You get max 5 points for getting pretty far, but not finishing the if/else or loop logic
   - You get max 3 points for main, libraries, variables, and reading input.
   - You get max 1 point for no clue but showing up!!!
3. When you finish, get a TA to check you off.

You MUST get checked off for the above before continuing!!!

You can complete this section individually or in pairs.

(3 pts total) Using an example to learn about Pass by Reference

Write a function called **get_sentence()** that gets the initial input string/sentence. Create a string object to read the string from the user, **#include <string>**. Since, a user can enter a single word or a string with spaces, you will need to use the getline() function to read the string/sentence from the user, i.e. **getline(cin, s).** Use this function to get the string/sentence from the user and print the contents of the string after making the call to this function. For example:

```
int main() {
   string sentence;

   sentence = get_sentence();
   cout << sentence << endl;

   return 0;
}
```

The prototype for this function is as follows. You need to determine pre-conditions and post-conditions for this function. The preconditions are anything you can say holds true about the function parameters before the function is called, and the postconditions are anything you can say holds true about the parameters, after the function executes.

string get_sentence();

Now, change the get_sentence() function so that it is a void function, which means you cannot assign the value returned from the function call to the sentence variable in main! Since the function is now a void function, it has to modify the string parameter directly, instead of returning information. You need to add an & to the string parameter.

void get_sentence(string &s);

The & in the parameter list tells us that this is a reference to string, and the string can be changed inside the function.

- Write the function without the & in the parameter listing, what do you notice?
- Write the function with the & in the parameter listing, what do you notice?

Including the **&** causes the string to be passed by reference. As you can see, this has a major impact on the way your code functions.

What are the preconditions and postconditions for **void get_sentence(string &s)?**