Lab 7

In order to get credit for this 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 work before the next lab. For extenuating circumstances, contact your lab TAs and the instructor.

Pair Programming: In this lab, you can choose a partner for pair programming. You must be checked off together. As with previous labs, you only need one computer for pair programming.

(6 pts total) Design First (3 pts), Then Write Code (3 pts)

To help you practice pass-by-reference and pass-by-pointer, you will write a short program that asks the user to enter a string (**get_string**()), makes a copy of the string, takes the copy and changes all non-space letters to dashes (**set_replace_string**()). The program then gets a letter from the user to search in the original string and replace the dashes in the copied string with the letter found, returning the number of letters found (**get_search_replace**()).

You should design first. You can have more functions, but you must have at least the three below with the EXACT function prototypes.

Each of your functions should have the proper function headers/descriptions

```
void get_string(string *);
void set_replace_string(string, string *);
int get_search_replace(char, string, string &);
```

Write the function headers/descriptions for each of the functions above, as well as all the functions you create. This includes information about parameters, return values, and pre/post conditions.

Design – Give as much detail as possible for the main function and all the functions above.

- Why do the function prototypes have the specific parameter types on them?
- Why do the functions have void or a return value?
- How do all the functions interact together?

Testing – Provide testing values with expected results.

- What do you plan to use as bad values? Good values?
- What do you expect to happen with bad values? Good values?

Get checked off by a TA before beginning to implement your design. This will help with logic and function-related mistakes**.**

(4 pts) Practice with Pointers and Valgrind

As discussed in class, it's not always easy to spot bugs that involve dynamic memory. For this part of the lab you will download some example code and use Valgrind to troubleshoot it.

Download the file:
https://web.engr.oregonstate.edu/~goinsj/resources/CS161/dynamic_memory_bugs.cpp

The code was intended to compute the average grade for a defined number of students. Each grade is supposed to be between 0 and 100. For some reason the computed average isn't correct. Use Valgrind and your troubleshooting skills to fix this code.

Valgrind usage: `valgrind a.out`

Once the code is working properly, the final line of the Valgrind output should say that there are 0 errors and the LEAK SUMMARY should show 0 bytes lost.