Part 4. (25 pts) Analyze Your Work (in a Word/text file, section "Part 4")

(A) (2 pts) As you worked to implement your Text Surgeon, you probably thought of new test cases after your Design Document was already submitted. List one here. (If you didn't think of any new test cases while implementing, create one now.)

When chosen to swap a character, choose a character than does not exist in the string and change it with another character. The program would output the same thing as the string because there is nothing to change.

(B) (9 pts) Try each of your 9 test cases (there are 9 now) and for each one, report whether or not the behavior is as expected. If not, state whether (1) your design has evolved and now the expected behavior is different (state what the new expected behavior is) or (2) this test helped you find (and fix) a bug in your program. 5 o If you didn't have 8 test cases in your Design Document, make them up now to allow you to get full credit here.

The test cases helped me to visualize and think about the limits of my program so that I know which range to take in and the data type and so on. Everything I thought of during the design phase is as I implemented in the code because I normally would prefer to start with coding instead of designing so I usually have the entire flow of the code in my head when I design, which meant there was no need to do an extra design.

(C) (5 pts) When you ran valgrind, did you find a memory leak? If so, copy and paste valgrind's message about the memory leak. If not, create a memory leak on purpose to generate a valgrind message for you to copy.

```
==9294==
==9294== HEAP SUMMARY:
==9294==     in use at exit: 1,028 bytes in 1 blocks
==9294==   total heap usage: 1 allocs, 0 frees, 1,028 bytes allocated
==9294==
==9294== LEAK SUMMARY:
==9294==    definitely lost: 1,028 bytes in 1 blocks
==9294==    indirectly lost: 0 bytes in 0 blocks
==9294==      possibly lost: 0 bytes in 0 blocks
==9294==    still reachable: 0 bytes in 0 blocks
==9294==         suppressed: 0 bytes in 0 blocks
==9294== Rerun with --leak-check=full to see details of leaked memory
==9294==
==9294== For counts of detected and suppressed errors, rerun with: -v
==9294== Use --track-origins=yes to see where uninitialised values come from
==9294== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
```

(D) (5 pts) How did you fix the memory leak you found? (If you created it on purpose, describe how you created it.) Either way, explain what was wrong with the C++ code. Fix the error before submitting your code.

The memory was leaked when it is an array of character is created in the heap and nothing was deleted off the heap when the program ends. To solve this, I have to delete the array of the pointer pointing to the heap; delete [] <pointer name>

(E) (4 pts) What was the hardest part of this assignment?

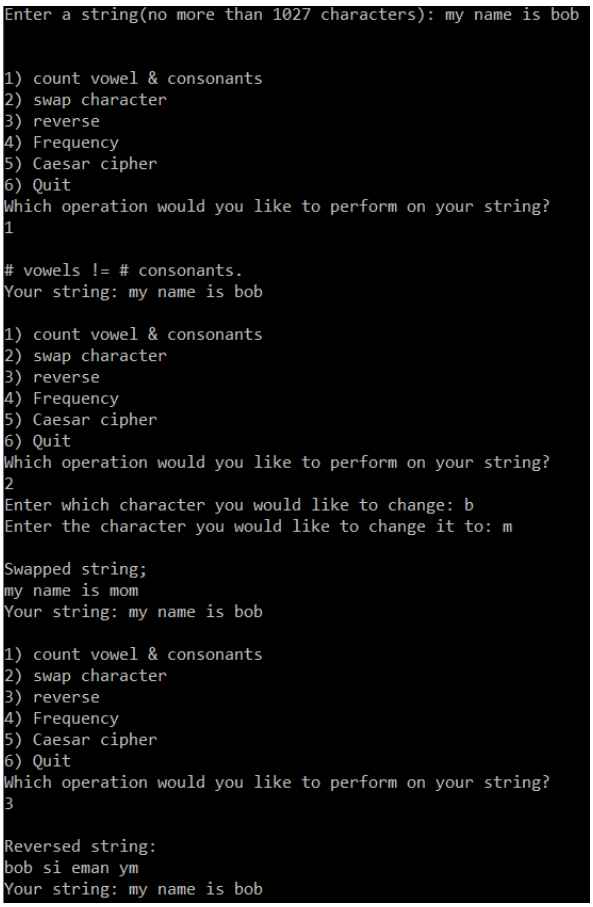
Determining where to return the pointer to and what to pass into the function, whether it is passing the pointer into a pointer parameter or passing in an address into a pointer parameter.

Part 5. Extra Credit (write in the Word/text file, section labeled "Part 5")

• (up to 2 pts) Implement all 4 of the operations described above (plus your custom operation), instead of only 3 (plus your custom operation.

• (up to 2 pts) Ask another person (friend, family member, roommate, coffee shop stranger) to try out your program. In "Part 5" of your file, copy/paste their full interaction

(program outputs and what they typed). Did the program crash or behave in a way you did not expect? Describe anything you found surprising.



```
1) count vowel & consonants
2) swap character
3) reverse
4) Frequency
5) Caesar cipher
6) Quit
Which operation would you like to perform on your string?
4
Enter up to 10 letters to find the frequency of in your string: mbkl
m) 2
b) 2
k) 0
l) 0

Your string: my name is bob

1) count vowel & consonants
2) swap character
3) reverse
4) Frequency
5) Caesar cipher
6) Quit
Which operation would you like to perform on your string?
5
Enter the caesar shift number you would like to encrypt your string with(-25 to 25): 20
gs hugy cm viv
Your string: my name is bob

1) count vowel & consonants
2) swap character
3) reverse
4) Frequency
5) Caesar cipher
6) Quit
Which operation would you like to perform on your string?
6
```

• (up to 2 pts) Describe one improvement you would recommend to make your program better (more robust, more entertaining, more attractive, anything).

If I had more time, I would implement error messages telling the user to do what I expect them to do exactly, like you entered a character that does not exist, or your string has only a space bar or only one character, then prompt them to enter again.

• (up to 2 pts) Add another operation to your Text Surgeon that counts the occurrences of a given word (prompting the user for this word too):

int count_word(char* s, char* word);

In this setting, a "word" is defined as a sequence of letters and or numbers not including

whitespace (space, tab, newline, etc.), punctuation mark, or other non-letter, nonnumber character. Results are not case-sensitive.

Example: User enters "Off we go, into the wild blue yonder!", then enters

word "off"; program prints:

1 occurrence of "off".

Example: User enters "The heart of the eye of the world", then enters word

"the"; program prints:

3 occurrences of "the".