CSci 1113: Introduction to C/C++ Programming for Scientists and Engineers Homework 3 Spring 2019

Due Date: Monday, Feb. 25 before 5:30pm.

Purpose: In this homework you get the chance to write more C++ programs. These programs will involve some of the basic C++ constructs including the string data type (a little), the char data type, selection statements, and complicated looping structures.

Instructions: This is an individual homework assignment. There are two problems worth 30 points each. Solve each problem below by yourself (unlike the labs, where you work collaboratively), and submit each solution as a separate C++ source code file. Here are a few important reminders:

- 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. See the collaborative rules on the class website for more information.
- Remember to follow carefully all naming conventions, submit the correct files through the class website by the deadline, follow the example input and output, and ensure your code runs on cselabs computers running the Linux operating system.
- Also remember to use good style.
- In each problem one or more test cases is given as an example; however, your problem will also be graded on other test cases which you will not know in advance. (One part of programming is making up your own test cases and testing your program on them.)

Problem A: Message from Tantive IV (30 points)

The rebellion needs your help! In mere moments the ship you are riding in will be overtaken by the imperial fleet. We must get an encrypted message (a cipher text) off the ship and to our resources on the ground. We need you to create a Caesar cipher that can convert plaintext messages to encrypted ciphertext messages, as well as decrypt ciphertext messages back into plain text.

A Caesar cipher is a mechanism of encryption where every letter in a message is rotated by a pre-agreed upon amount, thus scrambling the important message.

Example:

1. Take the English alphabet : ABCDEFGHIJKLMNOPQRSTUVWXYZ
2. Define how far to rotate: (+3 in this example)
3. The cipher is: DEFGHIJKLMNOPQRSTUVWXYZABC; where plaintext ‘A’ $\rightarrow$ ciphertext ‘D’, plaintext ‘B’ $\rightarrow$ ciphertext ‘E’, and so on.

4. Input a message in English: HELP
5. To encrypt, translate that message using the cipher. H $\rightarrow$ K, E $\rightarrow$ H, L $\rightarrow$ O, P $\rightarrow$ S
6. HELP $\rightarrow$ KHOS
7. To decrypt a ciphertext message, translate it back into English by reversing the process. For example, with rotation amount +3, KHOS $\rightarrow$ HELP.

Your task is to write a C++ program that will have:

1. Input:
   (a) Prompt the user to encrypt (e/E) or decrypt (d/D).
   (b) Prompt the user for an amount to rotate by (0 - 25).
      i. For example, 0 would mean the message remains clear text A $\rightarrow$ A
      ii. For example, 25 would mean that A $\rightarrow$ Z
   (c) Prompt the user for a message.
2. Output:
   (a) Decrypt or encrypt as appropriate and print out the translated message.
3. Error Conditions:
   (a) The program should prompt the user for all input, then after the prompt and input is complete:
      i. If the encryption/decryption mode is invalid, you must quit the program.
      ii. If the rotation is > 25 or < 0 you must quit the program.

Example 1 (user input is underlined):

```
Encrypt (e/E) or Decrypt (d/D)?
e
What is the cipher rotation value?
10
What is the message?
Help me Obi Wan
Translated Message:  ROVZWOYLSGKX
```

Example 2 (user input is underlined):

```
Encrypt (e/E) or Decrypt (d/D)?
d
What is the cipher rotation value?
17
What is the message?
mruvizjclbvjwrkyvi
```

```
Translated Message:  VADERISLUKESFATHER
```

Example 3 (user input is underlined):

```
Encrypt (e/E) or Decrypt (d/D)?
d
What is the cipher rotation value?
-42
What is the message?
badrotation
INVALID ROTATION...  TERMINATING
```

Some additional rules and guidance:

- The program must handle both encryption and decryption
  - Prompt the user for ‘e/E’ or ‘d/D’
    - If the user inputs anything other than that, you must print the error message:
    - `INVALID MODE...  TERMINATING`
    - Then exit the program.
    - `exit(1)`
- The USER must be able to specify the rotation amount (as a number between 0 and 25, inclusive).
- The application should do some validation on the rotation amount
  - IF the rotation amount is > 25 or < 0 you must print the error message:
  - `INVALID ROTATION...  TERMINATING`
  - Then exit the program.
  - `exit(1)`
- The application should also convert any lower case characters in the message to UPPERCASE
- The application should remove any non-alphabetic characters (a-z) from the message and not translate or print them.
  - E.g. → HELP ME PLEASE! → HELPMEPLEASE

A few tips:

- This program will require at least one loop, and might require several.
- You can loop through a string using a `for` loop, and use the function `.at(position)` to get the character value.
  - `string str = "CLIFFORD";`

  - `char ch = str.at(0)`
  - `ch` would equal ‘C’; // which has ASCII value 67

- You will likely find the ASCII table useful for figuring out how to convert from lower case to UPPERCASE letters, or for doing other needed operations for this program. Google “ASCII table” to find an online version of the table, or see Appendix 3 in the textbook for a hardcopy version.

- To read in a string that might include whitespace, use the `getline function`. One complication is that when using `cin` and `getline` you must make sure to flush the buffer. To make this easy on you, we have included some code to help you get started. It declares the 3 input variables and sets up the prompt. You will NOT need to modify the prompt, just starting your code below it.

When you are done, name the source code file `<username>_3A.cpp`. Here you replace `<username>` with your U of M email address; for example, if your email address is `smithx1234@umn.edu`, your file should be named `smithx1234_3A.cpp`. Then submit your program using the HW 3 Problem A submission link in Canvas.

Problem B: Here’s your sign (30 points)

A local rancher has asked you to help them come up with a plan for laying fencing on their property. They need you to print out a blueprint for how to lay the fencing where a ‘|’ represents a post, a ‘–’ represents a foot of fencing and a ‘S’ represents a single no trespassing sign. Your task write a C++ program that prompts the user for input, and then calculates and prints out (i) the fence blueprint and (ii) how many signs to order.

Rules:

- Your program must prompt for the following: the total number of posts, the number of feet between posts, and the minimum feet between signs.

- A sign may NOT be adjacent to a post or on a post.

- For any section between two posts, the number of feet of fence laid + the width of any signs(s) set must be equal to the feet between posts. There can be no ’missing’ feet between posts. There can be no extra feet between posts.

- The distance between signs is regulated by the minimum feet between signs. If possible the next sign must be this minimum number from the previous sign. If a sign cannot be located the minimum number, then the sign must be placed in the next allowable location. The width of every sign is 1 foot.

- No partial feet of fence may be laid. No partial signs may be set. All fence and signs is measured in whole feet.

- Posts have a 0 foot width. Posts shall not cause the length of the fence to increase.

- The first sign must be at least the minimum feet between signs from the initial post.

- The minimum feet between signs is the minimum needed feet of fence between signs.
  - Example: in the output `|--S--S--S-|` the feet between the first sign and the second sign is 2.

Example 1 (user input is underlined):

```
How many posts do you want to use?
5
How many feet between posts?
5
What is the minimum distance between signs (in feet)?
2
|--S--|-S---|-S---|-S---|
Total Signs:  4
```

Example 2 (user input is underlined):

```
How many posts do you want to use?
8
How many feet between posts?
10
What is the minimum distance between signs (in feet)?
16
|----------|------S---|----------|---S------|----------|-S--------|--------S-|
Total Signs:  4
```

Example 3 (user input is underlined):

```
How many posts do you want to use?
3
How many feet between posts?
20
What is the minimum distance between signs (in feet)?
60
|--------------------|--------------------|
Total Signs:  0
```

Tips:

- You will need several loops and if statements.
- You will need to track exactly how far along the fence you have gone.
- Consider building a string incrementally and printing the string out at the end of the program; this will allow you to use `cout`'s for debug statements.

Assumptions and further clarification:

- Assume that the fence in question does NOT form a circuit, (i.e., it does not end at its first post, but that the first and last posts are different), and that is is composed of individual fence segments (i.e., there is no overlapping of segments, no places the fence crosses itself, etc.).
- Assume all inputs will be > 0.
- It is possible that no signs could be set. See Example 3.

When you are done, name the source code file `<username>_3B.cpp`. Here you replace <username> with your U of M email address; for example, if your email address is `smithx1234@umn.edu`, your file should be named `smithx1234_3B.cpp`. Then submit your program using the HW 3 Problem B submission link in Canvas.