CSCI 201 – Computer Science 1

Homework Assignment 4

A program that counts frequencies of characters.
Due on Wednesday February 7

IMPORTANT The sample code for processing a text file can be found in `CourseInfo/Files/FileTest2.cpp` in `CourseFiles`.

Objectives. *The objectives of this assignment are:*
1. Learn how to process a text file using a `while` *loop C++.*
2. Process multiple files by nesting a `while` *inside a* `for` *loop.*

In this assignment, you will create a program that counts the frequencies of some characters and prints the result to an output file. For the current usage trends in English, the characters E, T, A, O, I, N, S, H and R are the most common ones. Your program will count the occurrences of each of these characters in the input file, and output the number of occurrences of each character, and also the total number of characters. The program should handle multiple files in a single execution. The program first prompts for the number of files to be processed and the name of the file where the results should be written; for each file it prompts for the name of the file, and prints the results to the specified output file. As an example, a sample run should look something like this:

```
 Welcome! How many files are you processing: 2
 Please input the name of the output file: Counts.txt
 Please input the name of the  file: FirstFile.txt
 FirstFile.txt has been processed.
 Please input the name of the  file: AnotherFile.txt
 AnotherFile.txt has been processed.
 Thank-you, Goodbye!
```

The output file (`Counts.txt`) should end up looking something like this:

```
File name: FirstFile.txt
Number of E's: 19
Number of T's: 15
Number of A's: 19
Number of O's: 9
Number of I's: 2
Number of N's: 7
Number of S's: 11
Number of H's: 4
Number of R's: 10
Total Number of characters: 237
-----------------------------------------------
File name: AnotherFile.txt
Number of E's: 16
Number of T's: 11
Number of A's: 13
Number of O's: 9
Number of I's: 2
Number of N's: 13
Number of S's: 11
Number of H's: 5
Number of R's: 11
Total Number of characters: 197
```

Some important points to keep in mind:

1. *Create data files in* `gedit or notepad++` *(not* `Word`*).* There should be no blank lines at the end, i.e., no carriage returns after the last character.
2. *First, write and test a program for one file only.* This program will have only one loop, a *while* loop. Once you have this done this correctly, you are ready to build a nested loop.
3. *Nest the while loop inside a for loop.* This is how we will build a program that can process multiple files. Read the notes below before you code and test this program.
4. *Clear the input file stream variable after each file.* This is done by inserting the statement `infile.clear()` immediately after `infile.close()`. This is needed because the same file stream is being used for all the input files.
5. *Choosing appropriate kinds of control structures.* Use a `for` loop that executes as many times as the number of files to be processed and an inner `while` loop that processes each file. Choose an appropriate branching construct; we want the code to look clean and easy to read.
6. *Counting both upper and lower case.* Remember to include both the upper and lower case letters when you count.
7. *Testing.* Tests should use text files that come from 2 or 3 different sources, and should include some for which the counts are independently verified. Report these results in a table.
8. *Dealing with bad input file names.* If the user enters a file name that does not exist, the program should not exit (via `return 0`), but simply move on to the next file, i.e., skip over the rest of the for loop for that iteration.

What to turn in. The following items must be uploaded to a folder named `Hwork4` within your CourseFiles folder.

- A script file (showing the source file, compilation and the test runs). Display the input files before each execution and the generated output file after each execution.
- A table showing the test results.
- **A user manual.** A good user manual should contain the following information in an itemized format and a friendly layout.

1. The purpose of the program.
2. Where to find the program and how to run it.
3. What the input should look like and what kind of output to expect. Use examples as needed.
4. Any special conditions and caveats.

- **Reflection.** What different kinds of looping and branching constructs did you use? Were these the most appropriate in each case? Explain. How did you deal with bad input file names?