Programming Assignment 4.4

**Assignment Description**

In this assignment, you'll be developing an application that takes a string of characters and counts how many times each of the alphabetic characters appears. Case doesn't matter for the counts, so the string aAa should yield a count of 3 for the letter A. Finally, the program should print, for each character that appears at least once in the phrase, the character and the count for that character.

**Motivation**

The letters of the alphabet appear in English prose with specific frequency; for example, the letter A appears far more often than the letter Q. One technique we can use to crack simple substitution encryption techniques is to count the frequency of letter occurrences in the encrypted message, then use information about their frequency in general use to try reasonable character substitutions based on those frequencies to try to crack the encrypted message. This approach works better on longer messages, and it's definitely a brute force cracking technique, but it's better than random guessing!

**Getting Started**

Download the appropriate zip file below for your OS and unzip the file somewhere on your computer.

**Windows Programming Assignment 4.4 Materials** ZIP File

**Mac Programming Assignment 4.4 Materials** ZIP File

Open the project in Visual Studio.

**Important:** You MUST only add code as indicated by the comments in that file. If you don't, you're virtually guaranteed to fail all the test cases in the automated grader.

**Requirements**

The code I’ve provided in the Program.cs file reads in a string of characters. You need to count how many occurrences of each letter are in the string and print that result.

Your solution must do the following:

* Count how many times each alphabetic character appears in the phrase, ignoring case (so A and a both add to the count of As)
* For each character that appears at least once in the string, print the character and the count for that character. You should have one space between each character count output, with all the character counts on a single line

For example, for the string aAaBbZ, your output should be the following on a single line (be sure to include a newline at the end):

**A3 B2 Z1**

**Note: The last character count in your output (Z1 for the example above) must be followed by a single space**. Trust me, it’s easier that way, so the automated grader assumes you do that!

We’d typically label our output to tell the user what the output means, but that will just confuse the automated grader. You must print ONLY the values described above in your output, with the values appearing on a single line with a single space between them.

**Helpful Hint 1**

The **Char** class provides useful functions for checking whether or not a character is a letter and converting a character to upper or lower case

**Helpful Hint 2**

You should use an array of 26 elements to store the counts for each of the characters, with element 0 holding the count of As, element 1 holding the count of Bs, and so on. Be sure to initialize all the elements to 0 before starting to count characters in the user phrase.

The Program.cs file I provided to you contains helper methods that you can use as you develop your solution. The **ConvertCharToIndex** method converts an upper case character (like A) to the index you should use in your counts array (for A, the corresponding index is 0). Make sure you pass in an upper case character to that method. The **ConvertIndexToChar** method converts an index to the corresponding character..

**Helpful Hint 3**

Just ignore non-alphabetic characters as you count the alpha characters in the phrase. They're not errors, they just don't contribute to the character counts.

**Running Your Code**

Because of the code I included to work with the automated grader on Coursera, when you run your program the command prompt window will open and it will sit there doing nothing. To make your code run, type in a string of 99 or fewer characters and press the <Enter> key; your code should then run so you can check your output. For example, your input could be:

**aAaBbZ**

You can actually run your code again if you want to by typing in a string of 99 or fewer characters and pressing the <Enter> key again. When you’re ready to stop running your code, type q (for quit).

Note: Don’t start your string with a q; that will make the code think you’re trying to quit.

Here's what running the code multiple times with different inputs should look like (remember, you're including a newline at the end of each of your output lines). The first line is the integer input, the second line is your output line, and so on:

**aAaBbZ**

**A3 B2 Z1**

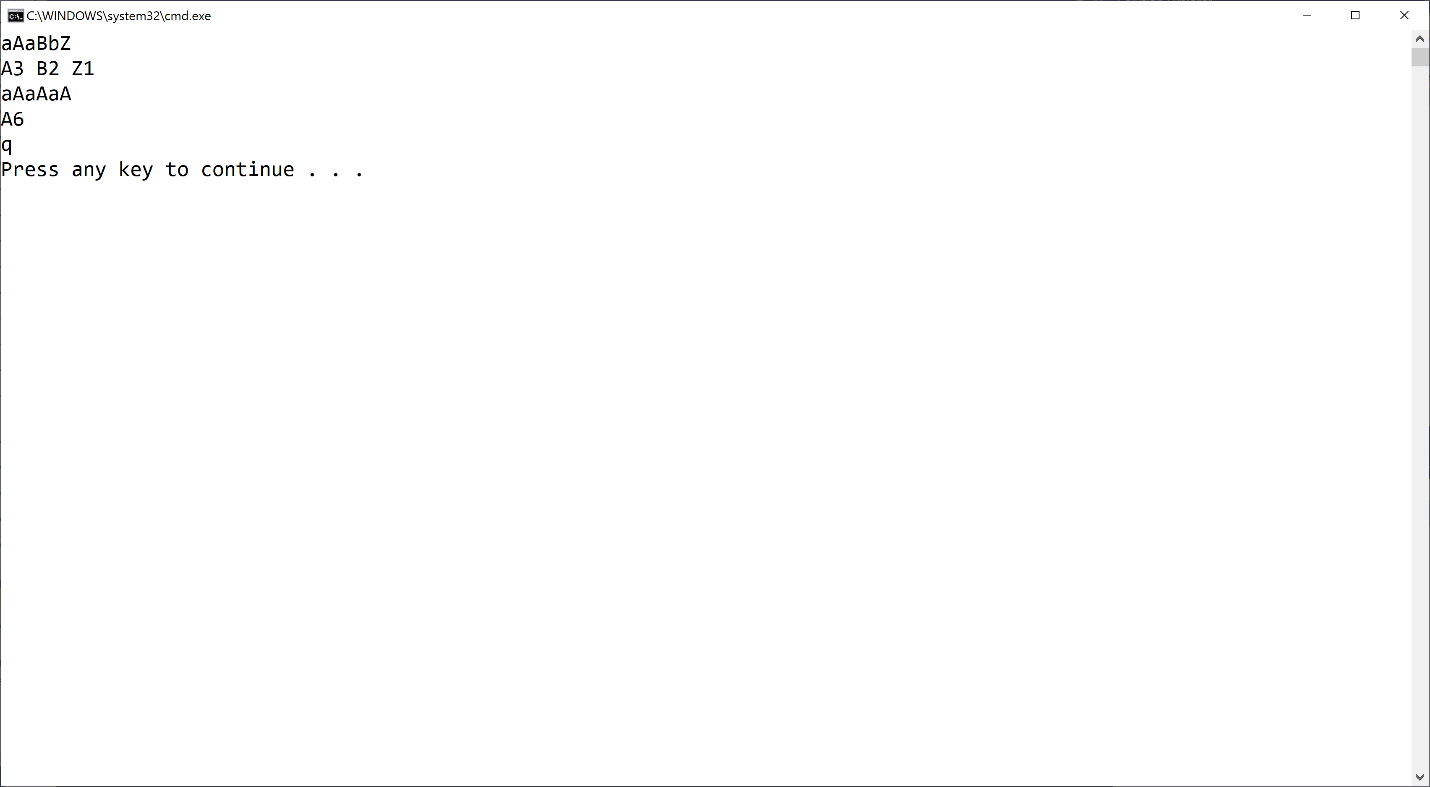
**aAaAaA**

**A6**

**q**

**Important Note:** The Coursera formatting makes it look like there's a blank line between each of the lines above, but there's not. The above output should be exactly 5 lines of output.

The image below shows my console window when I run the code multiple times as described above:



If your output doesn't match the image above EXACTLY (no extra words, characters, spaces, or blank lines) you'll fail all the test cases in the automated grader. **Remember, even though you can't see it above, each output line has a single space at the end.**

Mac users: You may or may not have an extra blank line after the q. That's fine.

**Test Case Inputs**

The autograder uses the following set of test case inputs to test your code (1 test case per line):

**aAaBbZ**

**aAaAaA**

**abcdefghijklmnopqrstuvwxyz**

**AbCdEfGhIjKlMnOpQrStUvWxYz**

**a1234567890b**

Each test case runs twice (for a total of 10 test cases), so if you're failing 2 test cases you're failing for one of the test case inputs above.

You should figure out what the expected results are for each test case input and make sure your code is generating those expected results in the required format before submitting your code for grading.

**Common Problems**

Historically, lots of people post about grading errors on this assignment (because their code is wrong). Here are a couple things you should check:

1. You have to start with the Visual Studio project I gave you in the assignment materials and add your code in the section indicated. The code I gave you includes the appropriate structure for the automated grader to work. If you don't do this, you'll almost definitely fail all the test cases in the automated grader.
2. I give you an example in this assignment (and all assignments) for what output you should get when you run the code multiple times. Be sure to try that example; if your code doesn't generate output exactly as shown in the example, your code isn't working properly

**Submitting Your Solution**

Go to the LMS and submit your project to **Assignment 4.4**