

CSC235 – Project 4, vowelCount

Assignment: Write a program that will search a user-supplied string of characters for the presence of characters in a list also supplied by the user.

Supplied: A folder containing a Visual Studio project that is set up for a MASM program.
A text file with an example input string.

Location: **/export/home/public/carelli/csc235/Projects/Project4**

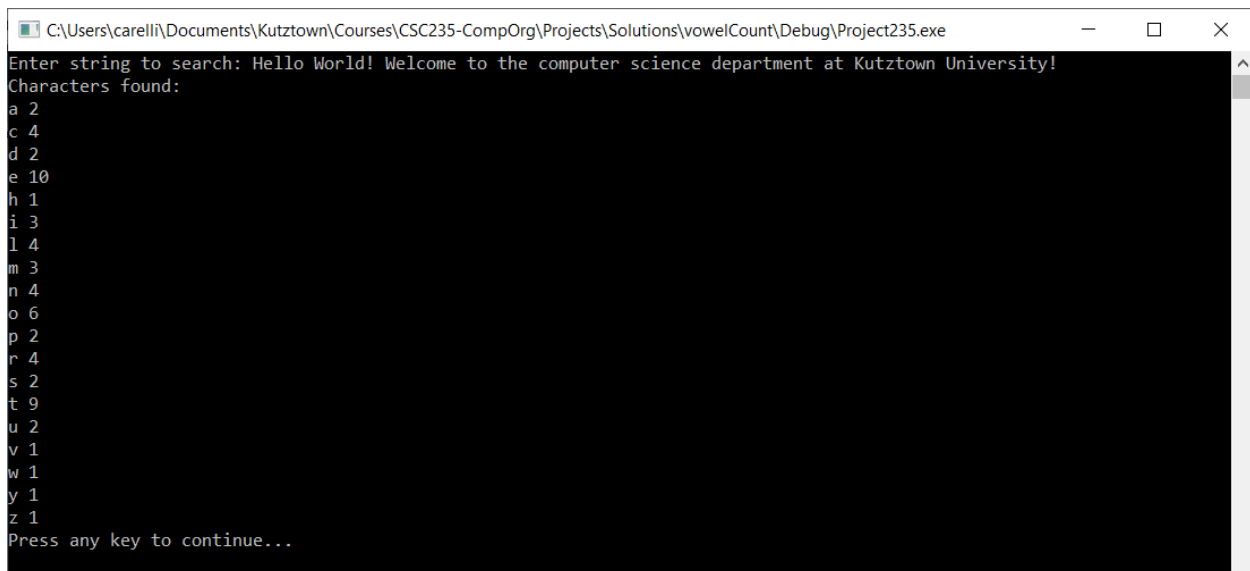
Deliverable: The Visual Studio solution/project folder containing the completed program.

Due: The programs MUST be turned in by the assigned date and time using D2L. Late submissions will, in fact, be rejected by D2L, resulting in a grade of zero.

Overview:

In this project, you will be introduced to the concept of writing a procedure in MASM. The task is to loop through a character string supplied by the user and test for the existence of lower-case letter in the string. A count should be stored in an array for each matching character. Finally, the program should, once more, loop through the lower-case letters, retrieve the count for each from memory, and, if the count is greater than zero, print out both the letter and the count. Skip over letters that did not appear in the string (zero count).

The image below illustrates how the program should operate. First, the user is prompted for a target string. The user types the string in - do not read it in from a file (you can copy and paste from the example file). The search is then conducted and the resulting output is a list of all letters found in the targeted string together with the number of occurrences.



```
C:\Users\carelli\Documents\Kutztown\Courses\CSC235-CompOrg\Projects\Solutions\vowelCount\Debug\Project235.exe
Enter string to search: Hello World! Welcome to the computer science department at Kutztown University!
Characters found:
a 2
c 4
d 2
e 10
h 1
i 3
l 4
m 3
n 4
o 6
p 2
r 4
s 2
t 9
u 2
v 1
w 1
y 1
z 1
Press any key to continue...
```

Requirements:

The main PROC will do the following:

- Prompt the user for a string. It can have up to 100 user-supplied characters (not including the terminating NULL character – extra characters should be ignored). It can have embedded whitespace, special characters, etc. – as shown in the example above.
- Read and store the *user string*.
- For each of the 26 lower-case letters (a-z). search the target string and count the number of occurrences of the given letter. This will be done in a procedure called *CountChar*, which is described below.
- For each lower-case letter (a-z), store the number of occurrences returned by *CountChar* in an array (26 values).
- Once every character in the *user string* has been processed and the number of occurrences for each lower-case letter has been stored in the array, output the letter and the count for letters whose count is non-zero.
- Note: do not store the letters themselves (a-z) in an array. It's wasteful and unnecessary.

You need to write the *CountChar* proc. As indicated, it will search a supplied string for a specific character (just one). It should count the number of times that character appears in the string and return that value. Both the string pointer and one character to search for should be supplied as inputs to the procedure. Do not search for more than one character in *CountChar*.

CountChar PROC:

- Inputs should be supplied to *CountChar* via registers as described in class and in Chapter 5 of the textbook.
- The output should be returned in a register.
- You will need to properly manage the stack so register data does not get corrupted either in *CountChar* or in main, which calls it.
- User input and output should be done using procedures supplied in the Irvine32 library. These are listed and described in Chapter 5 of the textbook.

Important Notes:

- *The looping over the 26 lower-case letters should occur in main*
- *The search for the specific target character in the user string should be done in CountChar*
- *Use symbolic constants to set fixed values*
- *Comment liberally!*
- *Follow code formatting guidelines!*
 - *Create a header comment section for your proc that lists:*
 - *Receives: what information is received, and in which registers*
 - *Returns: what information is returned and in which register*

Grading will be based on the following criteria:

1	10%	Turn in the <u>entire Visual Studio solution folder</u> , on time, to D2L
2	10%	Program must compile
3	10%	Use <i>Irvine32</i> procedures for user input and output <ol style="list-style-type: none">Display Prompt for the user string (see the example above)Output matched letters and count with an appropriate header message
4	20%	Store the count for each lower-case letter in memory <ol style="list-style-type: none">Array access and storage should be done in <i>main</i>
5	40%	Write the <i>CountChar</i> proc <ol style="list-style-type: none">Use registers for passing input parameters and the return valueInputs should include the pointer to the user string and the character to look forOutput should be the count (number of times the letter is found)Properly manage the stack so registers don't get corrupted when the routine is called
6	10%	Put your name in the header for the program where it says "Written By:" Use consistent formatting and comment liberally <ol style="list-style-type: none">With easily readable indentation and code labelingProvide a PROC header section describing information passed to/from the procedure and the registers usedThis is especially important in assembly language programming!!!