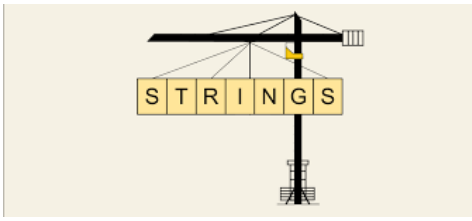
CST 180 C++ Programming
C-Strings Program Assignment (30 pts)



Purpose

To work with C++ c-strings and get additional experience with array processing.

Specifications

Declare an array to store a sentence (up to 80 characters). Prompt operator to enter a sentence (store in the array). Utilize the following functions to perform analysis on the input sentence.

FindLength	**Primary responsibility**: Return the length of the sentence. (Display length in main). HOW: Pass the sentence array to the function. Use a loop counter as a subscript to access each character: Initialize to zero and increment to locate the NULL character (end of string terminator '\0'). Return the counter value for display.
ConvertCaps	**Primary responsibility:** Convert the sentence to all capitals. (Display from main). HOW: Pass the sentence array to the function. Set up a loop looking for the NULL. Within the loop, examine each character of the string. If it is lower case (between 'a' and 'z'), convert to upper case by a simple math operation (see ASCII table).
CountLetters	**Primary responsibility:** Determines the number of letters contained in the sentence. (Display the results from main). How: Pass the sentence array to the function. Set up a loop to look for the NULL character. Inside the loop, use a decision construct to examine each character and determine whether the character is with the range of 'A-Z'.
CountVowels	**Primary responsibility:** Count the number of each vowel. # of A's, # of E's, etc. (Display the results from main). HOW: Pass the sentence array and 5 counters to the function. (Use referenece variables for the 5 counters because you can only return one thing via the return statement). Set up a loop to look for the NULL character. Inside the loop, use a decision construct to examine each character and determine which vowel counter (if any) should be incremented.
CountWords	**Primary responsibility:** Count the number of words in the sentence. (Display the results from main). HOW: Pass the sentence array to the function. Set up a loop looking for the NULL. Within the loop, examine each character of the string. Inside the loop, use a decision

	construct to examine each character and determine whether the character is a separator between words. Return the word count for display in main.
ReverseString	**Primary responsibility:** Copy the sentence in reverse order. (Display original sentence and reversed sentence from main). HOW: Pass the sentence array, an empty array, and the result from your FindLength function into this function. Initialize your loop counter to (string length - 1) and count down to zero. Use the loop counter as a subscript to access each character of the sentence from last to first. Assign that character to the empty array from first to last. (Note that you will need a second subscript for the second array.)

In addition to the above functions, you should also have your program determine the number of consonants that the string contains and output this value as well as the others. This is easily determined once you know the number of letters and the total number of vowels.

Design Requirement

Use planning documentation tools as needed to design your program. For turn-in, write pseudocode or a flowchart detailing the algorithm used to reverse the sentence.

Final Deliverables

Deliver the following in a single document :

- Cover page with assignment name, student name, and list of attachments
- Hierarchy chart that shows function calls
- Pseudocode or flowchart for ReverseString function
- Printed copy of source code
- Output screenshot(s)

Upload the document **AND** the .cpp source code file for grading into the appropriate dropbox.