Lab 07 : Strings and Recursion

I. Overview and Objectives

Strings in C are represented by arrays of characters. By putting a special character called the end-of-string or null character (represented by '\0'), the array of characters becomes a string that can be manipulated with specialized string functions.

Recursion is the process of calling a function by itself, until some specified condition is satisfied. It is used for repetitive computation (like finding factorial of a number) in which each action is stated in terms of previous result.

The learning objective of this lab is to practice working with strings in C and get an acquaintance with recursive solutions to problem solving and recursive functions in C

Reading and related topics: Course slides lessons 08 and 09. Book chapters 8 and 9.

II. Lab Tasks and Submission Guideline

Write complete C programs to solve the following problem. Save the code you wrote to solve it, together with the result of it in a report. Make sure you include enough comments in your code.

Copy and paste the source code (as text, not image), copy and paste the execution results of each of the outputs (high definition screenshot image) into your lab report. Save your report in .pdf format and submit it on D2L. You should submit your lab at the end of your lab session or soon after. In all cases it must be submitted before the deadline indicated in the D2L dropbox or it will not be accepted for marking.

Problem 1: A string is a palindrome if it reads the same from front to back as it does from back to front (e.g. "kayak", "rotator" and "noon" are palindromes). When determining whether an alphanumeric string is a palindrome, we often ignore spaces, punctuation and case in the string (e.g. "A man, a plan, a canal --Panama!" is also considered a palindrome).

Write a complete C program to read a string, echo it to the screen, determine whether it is a palindrome, and write a message indicating whether it is or isn't.

a) Your main function should prompt for the string, read the string, print it, call the ***clean*** function (to remove all spaces, punctuation, and turn uppercase letters into lowercase letters), print the new string, then call the ***reverse*** function (to place the characters into reverse order) and print the new string resulting. In the final step, you will compare the result of the reverse function with the original string and print a message if the string is a palindrome or not.

b) The ***clean*** function takes the original string and removes all characters that are not alphanumeric and converts letters to lowercase if necessary. Its prototype (header) is
void clean (char before[], char after[])

c) The ***reverse*** function places the characters of the original string in reverse order (ex: "abcde" becomes "edcba"). You must use a recursive solution for this function (no loops!). Its prototype (header) is
void reverse (char before[], char after[])

Use the following strings to test your program. The strings in green are palindromes, the ones in red are not.

Drab as a fool, aloof as a bard.
It ain't over till it's over
radar
When you come to a fork in the road, take it
Marge lets Norah see Sharon's telegram.

Have fun!