Exercise 5

All instructions regarding file names (hw5q1.c and hw5q2.c), as well as the students.txt file, the zip folder and compare your outputs to ours using DiffMerge is as usual for this exercise!

I prepared for you the general pattern of both questions on the files that are supplied with the exercise.

1) In this question you are required to implement two recursive functions (no loops are allowed!!).

a) A recursive function

void reverseCharacters(char *str, int n)

That has that gets a string str and its length n (not including the final \0 character) and replace the order of characters in the string, such that the first character becomes the last one, the second character becomes the character that comes before the last etc..

Pay attention:

- If the order of the string is odd, the character at index n/2 stays in place.
- The function does not print anything nor return any value.

For example, the word "Yummy" should turn to "ymmuY".

b) A recursive function

void PrintReverseWordsOrder(char *s, int i)

That gets a pointer to a string, representing sentence that has some number of words (separated by the space character ' '), and additional helper variable i, and prints the words appeared in s in reverse order, so that now the first word in the original sentence is the last word of the modified sentence etc. The order of characters in any word is not changed.
For example, the sentence "The book is on the table" should turn to "table the on is book The"

- You could assume that there is a single space between any two words and there are no spaces in the beginning nor the end of the string.
- You can modify the string.
- The value of the helper variable in the first call should be zero (as in the example).
- Printing the string could be done with %s in printf.
- You could assume that the maximum length of the sentence in part b is 30, and the maximum length of the word in part a to reverse is 10.
- To scan the sentence you could not use scanf with %s (which is not recommended in general!) because it contains spaces, you might use scanf char by char or preferably the getchar function.
- Hint: change spaces in the string to '\0'

For example, the output of the program on the sentence

The book is on the table

and the word Yummy

should be

```
C:\WINDOWS\system32\cmd.exe
Enter your sentence to reverse the order of words: The book is on the table
The reversed order of words of your sentence is: table the on is book The
Enter your word to reverse: Yummy
The reversed order of the string Yummy is: ymmuY
```
2) We define an array of integers to be "on and off sorted array" if it holds that

\[ a[i] \leq a[i+2] \text{ for any } 0 \leq i \leq n-3, \]

so the sub-array of even indices is sorted in a non-decreasing order, and similarly the sub-array of odd indices is sorted in a non-decreasing order. Write a function

\[
\text{int find}_x(\text{int } a[], \text{ int } n, \text{ int } x) \text{ that get an integer } x, \text{ as well as an "on and off sorted array" } a, \text{ and its size } n, \text{ and return a number } m \text{ such that } a[m]+a[m+1]=x. \text{ If no such number exist then find}_x \text{ should return -1.}
\]

- You can assume that the input is legal, in particular the array is an "on and off sorted array".
- The required running time is \( O(\log n) \).
- To make the exercise simpler we will assume that the size of the array is 10, no need to check that.

For example, the output of the program on the array

1 -6 5 9 7 13 12 14 19 25

and the number 44, should yield the following output:

```
The number 44 is the sum of two adjacent elements at indices 8 and 9 of the array.
```

And for the same array and number 19:

```
The number 19 is not accepted as the sum of two adjacent elements of the array.
```