CNG315 Homework 1 (Deadline: 31/10/2015)

In this homework, you are going to implement two functions.

PART 1: Make Anagram (50 pts.)

In this function, you will be given with two strings and the goal is to make them anagrams of each other. Two strings are anagrams of each other if they have same character set and same length. For example, strings "bacdc" and "dcbac" are anagrams, while strings "bacdc" and "dcbad" are not.

Given two strings (they can be of same or different length) find the minimum number of character deletions required to make two strings anagrams. Any characters can be deleted from any of the strings.

Input: Two strings that will be read from the text file. You will pass these strings to make_anagram function as parameters.

Output: Single (long) integer.

Function signature: long make_anagram(char *first_string, char *second_string);

Sample Input

```
cde
abc
```

Sample Output

```
4
```

In this example, a and b characters should be deleted from the second string. Following the same logic, d and e should be deleted from the first string. That makes 4 deletions in total to make the strings anagram.

This sample input is given only for visualization purposes. The function will receive inputs as parameters and return the result.

Complexity

Besides providing the correct solution, efficiency of your solution is also very important, and it will heavily affect your grade. I will test your functions' execution times and analyze the time complexity and the space complexity. A sample run on my computer may look like following:

```
Input Size: 100

Execution Time: 0 ms

Input Size: 10000

Execution Time: 0.4 ms

Input Size: 100000

Execution Time: 5 ms

Input Size: 500000

Execution Time: 31 ms

Input Size: 1000000

Execution Time: 75 ms
```

If your solution exceeds the execution time by margin of 10% in any of the inputs, you will lose 20 points. (Obtained execution times will be the average of 5 runs.) I may also follow a ranking-based grading system among students, where the fastest solution gets the highest grade. Try your hardest.

PART 2: Find Sum Parts (50 pts.)

In this function, you are given a sorted (long) integer array, and an (long) integer value. Your function should return the indexes of two integers from the given array that sum up to the given value.

Input: Sorted long integer array, size and sum that will be read from the text file. You will pass these to find_sum_parts function as parameters.

Output: Long integer array

Signature: long* find_sum_parts(long[]array, long size, long sum);

Sample Input

```
2 5 7 12 25 39 44 51 52 98 234

137
```

Sample Output

```
5 9
```

In this example 98 and 39 sum up to 137 and they reside in indexes 5 and 9.

This sample input is given only for visualization purposes. The function will receive inputs as parameters and return the result as a long integer array.

Complexity

Besides providing the correct solution, efficiency of your solution is also very important, and it will heavily affect your grade. I will test your functions' execution times and analyze the time complexity and the space complexity. A sample run on my computer may look like following:

```
Input Size: 100
Execution Time1: 0 ms
Execution Time2: 0 ms

Input Size: 10000
Execution Time1: 0 ms
Execution Time2: 0 ms

Input Size: 100000
Execution Time1: 0 ms
Execution Time2: 0.1 ms

Input Size: 500000
Execution Time1: 0 ms
Execution Time2: 2.6 ms

Input Size: 1000000
Execution Time1: 0.1 ms
Execution Time2: 3.2 ms
```

If your solution exceeds the first execution time by margin of 10% in any of the inputs, you will lose 15 points. If your solution exceeds the second execution time by margin of 10% in any of the inputs, you will lose 40 points. I may also follow a ranking-based grading system among students, where the fastest solution gets the highest grade.

Additional Information

- **Programming Language:** You must code your program in C++. Your submission will be compiled on my computer using Dev C++.
- **Library:** You are allowed to use any standard C++ library in your implementation including algorithm library.
- **Cheating:** In case of cheating, the university regulations will be applied.
- **Deadline:** 31/10/2015.

If you have any questions about the homework, please send me an e-mail.

Have fun,
Baris Engin Sonmez
Course TA