CSE-102 (A1)

Online on Pointer & String

Problem A [String]: Sherlock and Valid String

“You know my powers, my dear Watson, and yet at the end of three months I was forced to confess that I had at last met an antagonist who was my intellectual equal.”

A "valid" string is a string such that for all distinct characters in the string, each character occurs the same number of times.

For example, aabb is a valid string because the frequency of both characters a and b is 2, whereas aabbc is not a valid string because the frequency of characters a, b, and c is not the same.

Watson gives a string to Sherlock and asks him to remove some characters from the string such that the new string is a "valid" string.

Sherlock wants to know from you if it's possible to be done with less than or equal to one removal.

Constraints

- Maximum size of the string is 1000
- String contains lowercase letters only (a-z).

Sample Input

```
aabbcd
```

Sample Output

```
NO
```

Explanation

2 is the minimum number of removals required to make it a valid string. It can be done in following two ways:

Remove c and d to get aabb.
Or remove a and b to get abcd.

Problem B [Pointer]: Reverse Search

You are given two strings as input. Say, the first one is **s1** and the second one is **s2.** Your task is to find all indices of **s1** from where you will find **s2** in reverse order.

Consider the example:

s1 : "hello hello how are you?"

s2 : "olle"

The reverse string of **s2** is "ello". So, there are 2 indices in **s1**, 1 and 7 from where "ello" is found. (marked with yellow color).

The are some restrictions, as we expect you to use the core concepts of pointer:

- You cannot use [] operator for indexing
- You are not allowed to reverse any string, neither using library function nor using your own code. You must implement your program so that it searches **s2** in reverse order
- Only 2 strings (**char*)** can be used.One to strore **s1** and another to store **s2**. No more temporary string is allowed.
- You can only use these library functions:
  malloc(), realloc(), free(), printf(), scanf(), strlen(), gets()
  Using other library functions are strictly prohibited.
  [Note: malloc, realloc and free requires stdlib.h]

You may use as many integers as required [To store length, keep track of index, etc].

Problem C [Bonus]: Longest Common Substring

In computer science, the **longest common substring** is the longest string that is a substring of two or more strings.

Given two strings 'X' and 'Y'. Find the longest common substring and its length.

Input : X = "GeeksforGeeks", y = "GeeksQuiz"
Output : The longest common substring is "Geeks" and is of length 5.

Input : X = "abcdxyz", y = "xyzabcd"
Output : The longest common substring is "abcd" and is of length 4.

Input : X = "zxabcdezy", y = "yzabcdezx"
Output: The longest common substring is "abcdez" and is of length 6.