Sherlock and the Valid String

by darkshadows

Problem | Submissions | Leaderboard | Discussions | Editorial

Sherlock considers a string, s, to be *valid* if either of the following conditions are satisfied:

1. All characters in s have the same exact frequency (i.e., occur the same number of times). For example, $s = $ **"aabbcc"** is valid, but $s = $ **"baacdd"** is not valid.
2. Deleting exactly 1 character from s will result in all its characters having the same frequency. For example, $s = $ **"aabbccc"** and $s = $ **"aabbc"** are valid because all their letters will have the same frequency if we remove 1 occurrence of `c`, but $s = $ **"abcccc"** is not valid because we'd need to remove 3 characters.

Given s, can you determine if it's valid or not? If it's valid, print `YES` on a new line; otherwise, print `NO` instead.

Input Format

A single string denoting s.

Constraints

- $1 \leq |s| \leq 10^5$
- String s consists of lowercase letters only (i.e., `[a-z]`).

Output Format

Print `YES` if string s is valid; otherwise, print `NO` instead.

Sample Input 0

```
aabbcd
```

Sample Output 0

```
NO
```

Explanation 0

We would need to remove two characters from $s = $ **"aabbcd"** to make it valid, because `a` and `b` both have a frequency of 2 and `c` and `d` both have a frequency of 1. This means s is *invalid* because we'd need to remove more than 1 character to make all its letters have the same frequency, so we print `NO` as our answer.

Submissions: 17539
Max Score: 35
Difficulty: Medium

Rate This Challenge:

More

Current Buffer (saved locally, editable) Java 8

```java
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    static String isValid(String s){

        Map<Character,Integer> map = new HashMap<Character,Integer>();

        for(int i = 0 ; i < s.length() ; i++){

            if(map.containsKey(s.charAt(i))){
                map.put(s.charAt(i),map.get(s.charAt(i))+ 1);
            }
            else{
                map.put(s.charAt(i),1);
            }
        }

        ArrayList<Integer> list = new ArrayList<>(map.values());
        Collections.sort(list);


        if(list.size() == 2 && Math.abs(list.get(0) - list.get(1)) <= 1){
            return "YES";
        }

        if(list.get(0) == list.get(list.size() - 1)){
            return "YES";
        }
        else if((list.get(0) == list.get(1) && list.get(0) == list.get(list.size() - 2)) ||
                    ((list.get(list.size() - 1) == list.get(list.size() - 2)) && (list.get(1) == list.get(list.size() - 2)))){
            return "YES";
        }

        return "NO";

    }

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s = in.next();
        String result = isValid(s);
        System.out.println(result);
    }
}


```

Line: 50 Col: 1

Upload Code as File Test against custom input Run Code Submit Code