6078. Rearrange Characters to Make Target String

My Submissions (/contest/weekly-contest-295/problems/rearrange-characters-to-make-target-string/submissions/)

Back to Contest (/contest/weekly-contest-295/)

You are given two **0-indexed** strings `s` and `target`. You can take some letters from `s` and rearrange them to form new strings.

Return *the **maximum** number of copies of* `target` *that can be formed by taking letters from* `s` *and rearranging them.*

User Accepted:	0
User Tried:	1
Total Accepted:	0
Total Submissions:	1
Difficulty:	Easy

Example 1:

```
Input: s = "ilovecodingonleetcode", target = "code"
Output: 2
Explanation:
For the first copy of "code", take the letters at indices 4, 5, 6, and 7.
For the second copy of "code", take the letters at indices 17, 18, 19, and 20.
The strings that are formed are "ecod" and "code" which can both be rearranged into "code".
We can make at most two copies of "code", so we return 2.
```

Example 2:

```
Input: s = "abcba", target = "abc"
Output: 1
Explanation:
We can make one copy of "abc" by taking the letters at indices 0, 1, and 2.
We can make at most one copy of "abc", so we return 1.
Note that while there is an extra 'a' and 'b' at indices 3 and 4, we cannot reuse the letter 'c' at index 2, so we cannot make
```

Example 3:

```
Input: s = "abbaccaddaeea", target = "aaaaa"
Output: 1
Explanation:
We can make one copy of "aaaaa" by taking the letters at indices 0, 3, 6, 9, and 12.
We can make at most one copy of "aaaaa", so we return 1.
```

Constraints:

- `1 <= s.length <= 100`
- `1 <= target.length <= 10`
- `s` and `target` consist of lowercase English letters.

JavaScript

```
const counter = (a_or_s) => { let m = new Map(); for (const x of a_or_s) m.set(x, m.get(x) + 1 || 1); return m; };
const removeOneOrManyMap = (m, x, cnt = 1) => { let occ = m.get(x); occ > cnt ? m.set(x, occ - cnt) : m.delete(x); };

const rearrangeCharacters = (s, t) => {
    let ms = counter(s), mt = counter(t), res = 0;
    while (1) {
        let ok = true;
        for (const [c, occ] of mt) {
            let socc = ms.get(c) || 0;
            if (socc < occ) {
               ok = false;
               break;
            } else {
                removeOneOrManyMap(ms, c, occ)
            }
        }
        if (!ok) {
            break;
```

```
        } else {
            res++;
        }
    }
    return res;
};
```

Custom Testcase | Use Example Testcases

Run | Submit

Submission Result: Accepted (/submissions/detail/709480850/) | More Details > (/submissions/detail/709480850/)

Share your acceptance!

1

Copyright © 2022 LeetCode