6167. Check Distances Between Same Letters

My Submissions (/contest/weekly-contest-309/problems/check-distances-between-same-letters/submissions/) Back to Contest (/contest/weekly-contest-309/)

You are given a **0-indexed** string `s` consisting of only lowercase English letters, where each letter in `s` appears **exactly twice**. You are also given a **0-indexed** integer array `distance` of length `26`.

Each letter in the alphabet is numbered from `0` to `25` (i.e. `'a' -> 0`, `'b' -> 1`, `'c' -> 2`, ..., `'z' -> 25`).

In a **well-spaced** string, the number of letters between the two occurrences of the i^{th} letter is `distance[i]`. If the i^{th} letter does not appear in `s`, then `distance[i]` can be **ignored**.

Return `true` *if* `s` *is a* ***well-spaced*** *string, otherwise return* `false`.

User Accepted:	213
User Tried:	241
Total Accepted:	213
Total Submissions:	244
Difficulty:	Easy

Example 1:

```
Input: s = "abaccb", distance = [1,3,0,5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
Output: true
Explanation:
- 'a' appears at indices 0 and 2 so it satisfies distance[0] = 1.
- 'b' appears at indices 1 and 5 so it satisfies distance[1] = 3.
- 'c' appears at indices 3 and 4 so it satisfies distance[2] = 0.
Note that distance[3] = 5, but since 'd' does not appear in s, it can be ignored.
Return true because s is a well-spaced string.
```

Example 2:

```
Input: s = "aa", distance = [1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
Output: false
Explanation:
- 'a' appears at indices 0 and 1 so there are zero letters between them.
Because distance[0] = 1, s is not a well-spaced string.
```

Constraints:

- `2 <= s.length <= 52`
- `s` consists only of lowercase English letters.
- Each letter appears in `s` exactly twice.
- `distance.length == 26`
- `0 <= distance[i] <= 50`

JavaScript

```
const ord = (c) => c.charCodeAt();
const counter_value_in_indexA_in = (a_or_s) => { let m = new Map(); let n = a_or_s.length; for (let i = 0; i < n; i++) {
if (!m.has(a_or_s[i])) m.set(a_or_s[i], []); m.get(a_or_s[i]).push(i); } return m; };

const checkDistances = (s, d) => {
    let m = counter_value_in_indexA_in(s);
    for (const [c, a] of m) {
        let [i, j] = a, expectDis = j - i - 1;
        let dis = d[ord(c) - 97];
        if (dis != expectDis) return false;
    }
    return true;
};
```

Custom Testcase Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/790947445/) More Details (/submissions/detail/790947445/)

Share your acceptance!