424. Longest Repeating Character Replacement

Add to List

Description (?tab=Description) | Submission (?tab=Submission) | Solutions (?tab=Solutions)

Total Accepted: **7986** Total Submissions: **19687** Difficulty: **Medium** Contributors: **Admin**

Notes

Given a string that consists of only uppercase English letters, you can replace any letter in the string with another letter at most *k* times. Find the length of a longest substring containing all repeating letters you can get after performing the above operations.

Note:

Both the string's length and *k* will not exceed 10^4.

Example 1:

```
Input:
s = "ABAB", k = 2

Output:
4

Explanation:
Replace the two 'A's with two 'B's or vice versa.
```

Example 2:

```
Input:
s = "AABABBA", k = 1

Output:
4

Explanation:
Replace the one 'A' in the middle with 'B' and form "AABBBBA".
The substring "BBBB" has the longest repeating letters, which is 4.
```

Hide Company Tags | Pocket Gems (/company/pocket-gems/)

Hide Similar Problems | (H) Longest Substring with At Most K Distinct Characters (/problems/longest-substring-with-at-most-k-distinct-characters/)

Have you met this question in a real interview? Yes No

Discuss (https://discuss.leetcode.com/category/553) | Pick One (/problems/random-one-question/) | Editorial Solution

C++

```cpp
class Solution {
public:

    ///https://discuss.leetcode.com/topic/63448/consise-python-sliding-window
    // The basic idea is that one character should be fixed and other k characters are replaced to that fixed character.
    // The fixed character is choosen by the *max_element(v.begin(), v.end()) which has the highest frequency.
    // len - *max_element(v.begin(), v.end()) are the rest other character's total frequency which should be replaced.
    // if len - *max_element(v.begin(), v.end()) > k, we need to move i++ to shrink the window so that k replacement works

    int characterReplacement(string s, int k) {
        vector<int> v(26);
        int i = 0, j = 0;
        while (j < s.length()) {
            v[s[j] - 'A']++;
            int len = j - i + 1;
            if ((len - *max_element(v.begin(), v.end())) > k) {
                v[s[i++]-'A']--;
            }
            j++;
        }
        return j-i;
    }

    //  TLE solution T_T
```

Send Feedback (mailto:admin@leetcode.com?subject=Feedback)

```
25      // int helper(string s, int i, int k) {
26      //     if(i >= s.length() || k < 0) return 0;
27      //     //Don't replace s[i]
28      //     int t = k, j = i+1;
29      //     while(j < s.length() && (s[j] == s[i] || t > 0)) {
30      //         if(s[j] != s[i]) t--;
31      //         j++;
32      //     }
33      //     int maxlen = j-i;
34      //     return maxlen;
35      // }
36
37      // int characterReplacement(string s, int k) {
38      //     int N = s.length();
39      //     int maxlen = 0;
40      //     for(int i = 0; i < N; ++i) {
41      //         int len = max(helper(s, i, k), 1 + helper(s, i+1, k-1));
42      //         maxlen = max(maxlen, len);
43      //     }
44      //     return maxlen;
45      // }
46  };
```

Notes

Custom Testcase

Contribute Testcase

Run Code | Submit Solution