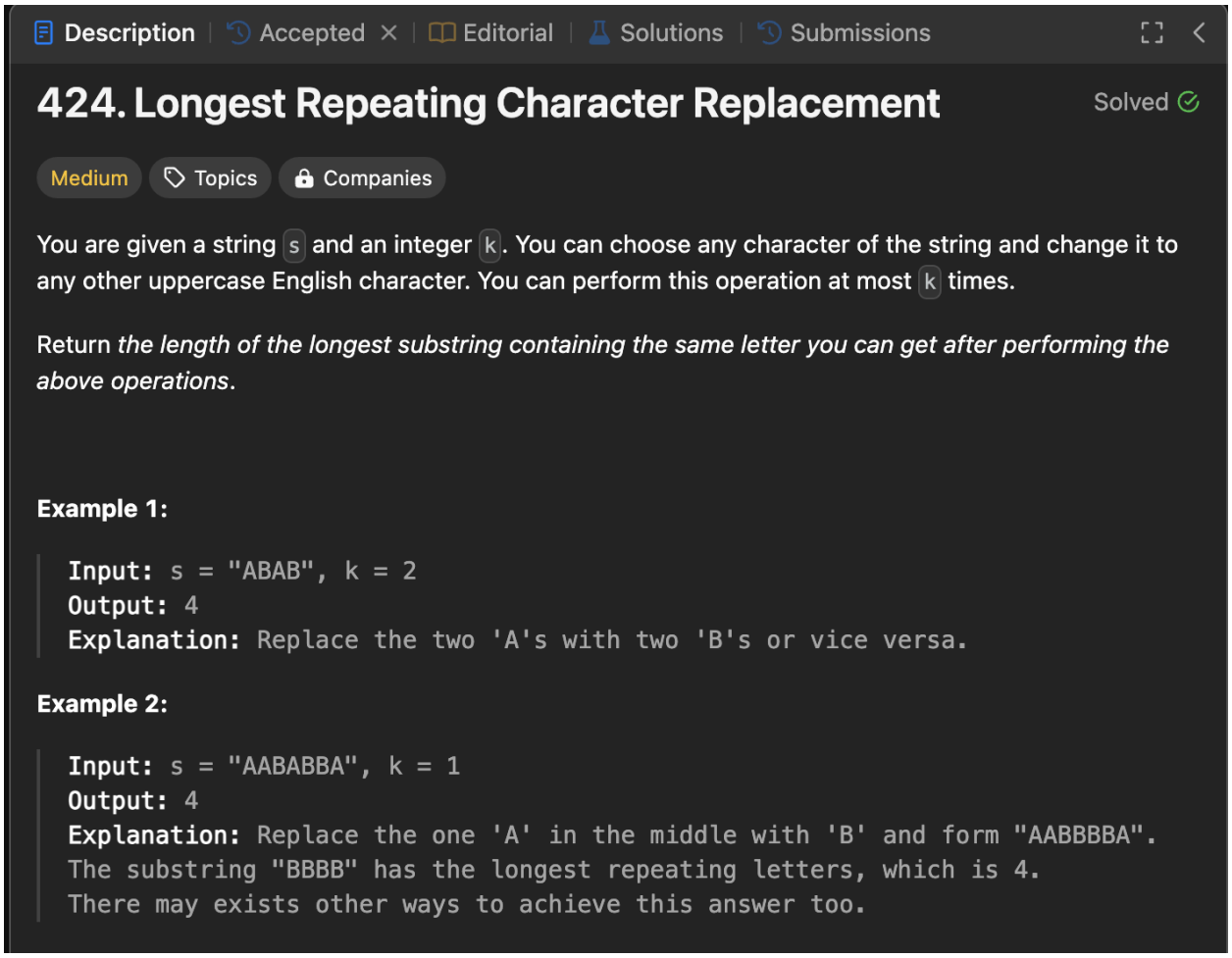
7: Leetcode 424. Longest Repeating Character Replacement

⇒

https://leetcode.com/problems/longest-repeating-character-replacement/description/?envType=list&envId=xlep8di5

Description



Description | Accepted | Editorial | Solutions | Submissions

424. Longest Repeating Character Replacement

Solved

Medium | Topics | Companies

You are given a string `s` and an integer `k`. You can choose any character of the string and change it to any other uppercase English character. You can perform this operation at most `k` times.

Return *the length of the longest substring containing the same letter you can get after performing the above operations*.

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.
There may exists other ways to achieve this answer too.
```

B - Code:

better:

```python
class Solution:
    def characterReplacement(self, s: str, k: int) -> int:
        longest,left = 0, 0
        sCount = {}
        maxCount = 0
        for right in range(len(s)):
```

```
            sCount[s[right]] = sCount.get(s[right],0) + 1
            maxCount = max(maxCount, sCount[s[right]])
            while (right - left + 1) - maxCount > k:
                sCount[s[left]] -= 1
                left +=1
            longest = max(longest, (right - left + 1))
        return longest
```

Mine:

```
class Solution:
    def characterReplacement(self, s: str, k: int) -> int:
        longest,left = 0, 0
        sCount = [0] * 26

        for right in range(len(s)):
            sCount[ord(s[right])-ord('A')] += 1
            while (right - left + 1) - max(sCount) > k:
                sCount[ord(s[left])-ord('A')] -= 1
                left +=1
            longest = max(longest, (right - left + 1))
        return longest
```

C- Notes

classmate
Date
Page

7] 424. Longest repeating character replacement:

→ A] Algo:

•) keep a count of almost the max char is present in the window.

•) A A B → winLen = r-l+1

A = [3, 1, 0, 0, ... 0]
0 1 2 3 25
(A) (B) (C) ... (Z)

∴ winLen - max(A) > k

3 - 2 > k if k=0
1 > k → then invalid.

•) If window valid, i.e. we have k (not 0), then we can replace the wrong diff char and make it valid & store its length.

•) If invalid, i.e. no more k (replacement) then remove count of left index & left ++.

B] Code:

→ 1) Initialize left, longest,
" sCount = [0]*26

2) For right in range (len(s)):

* sCount [ord(s[r]) - ord('A')] += 1

while (r-l+1 > k):

remove s[l] count (-=1) Invalid

l++.

3) longest = max(longest, r-l+1).

→ For better running time:

use .get to get char count.

sCount = {}

sCount [s[right]] = sCount.get(s[right], 0) + 1