←(/) Explore(/explore/) Problems(/problemset/all/) Interview ^New Contest^(/contest/) Discuss(/discuss/) 🏪 Store ☆ Premium (/subscribe?ref=nb_npl) 🔔 ♢ 0 **(/problems/shortest-bridge/)** 👤

6439. Minimum String Length After Removing Substrings

My Submissions (/contest/weekly-contest-346/problems/minimum-string-length-after-removing-substrings/submissions/)

Back to Contest (/contest/weekly-contest-346/)

You are given a string `s` consisting only of **uppercase** English letters.

You can apply some operations to this string where, in one operation, you can remove **any** occurrence of one of the substrings `"AB"` or `"CD"` from `s`.

Return *the **minimum** possible length of the resulting string that you can obtain*.

Note that the string concatenates after removing the substring and could produce new `"AB"` or `"CD"` substrings.

User Accepted:	10
User Tried:	16
Total Accepted:	10
Total Submissions:	16
Difficulty:	Easy

Example 1:

```
Input: s = "ABFCACDB"
Output: 2
Explanation: We can do the following operations:
- Remove the substring "ABFCACDB", so s = "FCACDB".
- Remove the substring "FCACDB", so s = "FCAB".
- Remove the substring "FCAB", so s = "FC".
So the resulting length of the string is 2.
It can be shown that it is the minimum length that we can obtain.
```

Example 2:

```
Input: s = "ACBBD"
Output: 5
Explanation: We cannot do any operations on the string so the length remains the same.
```

Constraints:

- `1 <= s.length <= 100`
- `s` consists only of uppercase English letters.

JavaScript ▾   ⟨⟩ ⟳ ⚙

```javascript
 1▾ const minLength = (s) => {
 2      let pre = s.length;
 3▾     while (1) {
 4          let used = new Set(), next = '';
 5▾         for (let i = 0; i + 1 < s.length;) {
 6              let cur = s[i] + s[i + 1];
 7▾             if (cur == 'AB' || cur == 'CD') {
 8                  used.add(i);
 9                  used.add(i+1);
10                  i += 2;
11▾             } else {
12                  i++;
13              }
14          }
15▾         for (let i = 0; i < s.length; i++) {
16              if (used.has(i)) continue;
17              next += s[i];
18          }
19▾         if (next.length == pre) {
20              break;
21▾         } else {
22              s = next;
23              pre = s.length;
24          }
25      }
26      return s.length
```

```
27  };
```

☐ **Custom Testcase**   (Use Example Testcases)

● Run   ☁ Submit

Submission Result: Accepted (/submissions/detail/954223697/) ❓   (More Details ❯ (/submissions/detail/954223697/))

Share your acceptance!

Copyright © 2023 LeetCode

Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)

🇺🇸 United States (/region)