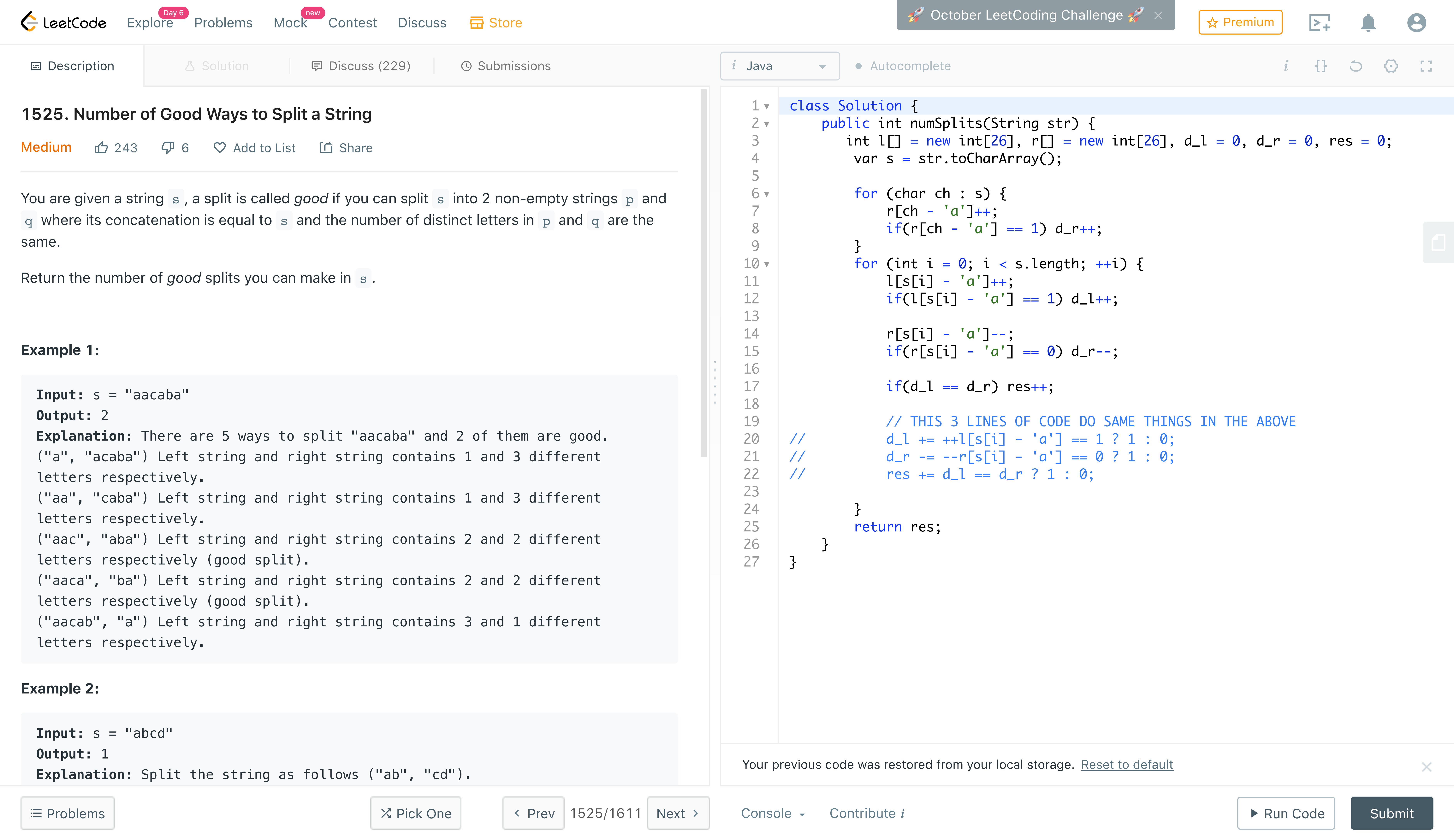

1525. Number of Good Ways to Split a String

Medium 243 6 Add to List Share

You are given a string `s`, a split is called *good* if you can split `s` into 2 non-empty strings `p` and `q` where its concatenation is equal to `s` and the number of distinct letters in `p` and `q` are the same.

Return the number of *good* splits you can make in `s`.

Example 1:

```
Input: s = "aacaba"
Output: 2
Explanation: There are 5 ways to split "aacaba" and 2 of them are good.
("a", "acaba") Left string and right string contains 1 and 3 different
letters respectively.
("aa", "caba") Left string and right string contains 1 and 3 different
letters respectively.
("aac", "aba") Left string and right string contains 2 and 2 different
letters respectively (good split).
("aaca", "ba") Left string and right string contains 2 and 2 different
letters respectively (good split).
("aacab", "a") Left string and right string contains 3 and 1 different
letters respectively.
```

Example 2:

```
Input: s = "abcd"
Output: 1
Explanation: Split the string as follows ("ab", "cd").
```

```java
class Solution {
    public int numSplits(String str) {
       int l[] = new int[26], r[] = new int[26], d_l = 0, d_r = 0, res = 0;
        var s = str.toCharArray();

        for (char ch : s) {
            r[ch - 'a']++;
            if(r[ch - 'a'] == 1) d_r++;
        }
        for (int i = 0; i < s.length; ++i) {
            l[s[i] - 'a']++;
            if(l[s[i] - 'a'] == 1) d_l++;

            r[s[i] - 'a']--;
            if(r[s[i] - 'a'] == 0) d_r--;

            if(d_l == d_r) res++;

            // THIS 3 LINES OF CODE DO SAME THINGS IN THE ABOVE
//          d_l += ++l[s[i] - 'a'] == 1 ? 1 : 0;
//          d_r -= --r[s[i] - 'a'] == 0 ? 1 : 0;
//          res += d_l == d_r ? 1 : 0;

        }
        return res;
    }
}
```

Your previous code was restored from your local storage. Reset to default