ref=nb\_npl)





# 5869. Maximum Product of the Length of Two Palindromic Subsequences

My Submissions (/contest/weekly-contest-258/problems/maximum-product-of-the-length-of-two-palindromic-subsequences/submissions/)

Back to Contest (/contest/weekly-contest-258/)

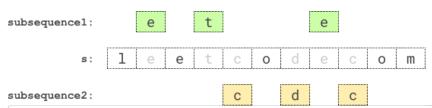
Given a string s, find two disjoint palindromic subsequences of s such that the product of their lengths is maximized. The two subsequences are disjoint if they do not both pick a character at the same index.

Return the maximum possible product of the lengths of the two palindromic subsequences.

A subsequence is a string that can be derived from another string by deleting some or no characters without changing the order of the remaining characters. A string is palindromic if it reads the same forward and backward.

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Medium

## Example 1:



Input: s = "leetcodecom"

Output: 9

**Explanation:** An optimal solution is to choose "ete" for the 1<sup>st</sup> subsequence and "cdc" for the 2<sup>nd</sup> subsequence. The product of their lengths is: 3 \* 3 = 9.

#### Example 2:

Input: s = "bb"

Output: 1

Explanation: An optimal solution is to choose "b" (the first character) for the 1st subsequence and "b" (the sec The product of their lengths is: 1 \* 1 = 1.

## Example 3:

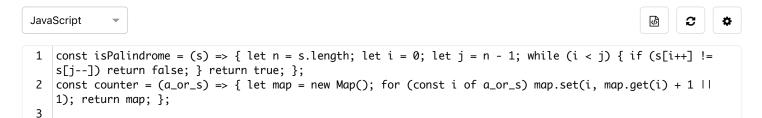
Input: s = "accbcaxxcxx"

**Output:** 25

**Explanation:** An optimal solution is to choose "accca" for the 1<sup>st</sup> subsequence and "xxcxx" for the 2<sup>nd</sup> subsequence The product of their lengths is: 5 \* 5 = 25.

### Constraints:

- 2 <= s.length <= 12
- s consists of lowercase English letters only.



```
4 \vee const maxProduct = (s) => {
 5
        let cnt = counter(s);
 6
         // let se = new Set();
 7
         let a = [];
 8
         let n = s.length;
 9
         let N = 2 ** n;
10
         // pr(n, N)
11 ▼
         for (let i = 0; i < N; i++) {
12
             let sub = '';
13
             let idx = new Set();
14 ▼
             for (let j = 0; j < n; j++) {
                 if (i & (1 << j)) {
15 ▼
16
                      sub += s[j];
17
                      idx.add(j);
18
19
             }
20
             // pr(sub)
21 •
             if (isPalindrome(sub)) {
22
                 a.push([sub, idx]);
23
             }
24
         }
25
         // pr(se, cnt);
26
         // let a = [...se];
27
         // a.sort((x, y) \Rightarrow y - x);
28
         // pr(a);
29
         let an = a.length;
30
         let res = 0;
         for (let i = 0; i < an; i++) {
31 ▼
32 ▼
             for (let j = i + 1; j < an; j++) {
33 ▼
                 if (isDisjoint(a[i][0], a[j][0], a[i][1], a[j][1])) {
34
                      // pr(a[i][0], a[j][0], a[i][0].length, a[j][0].length)
35
                      let len = a[i][0].length * a[j][0].length;
36
                      res = Math.max(res, len);
37
                 }
38
             }
39
         }
40
         return res;
41
    };
42
43 \vee \text{const isDisjoint} = (s, t, is, it) \Rightarrow \{
         for(const i of is) {
44 ▼
45
             if (it.has(i)) return false;
46
47
         return true;
48
    };
```

☐ Custom Testcase

Use Example Testcases

♠ Submit

Run

Submission Result: Accepted (/submissions/detail/553425509/) ?

More Details ➤ (/submissions/detail/553425509/)

Share your acceptance!

Copyright @ 2021 LeetCode

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

Privacy Policy (/privacy)



United States (/region)