

5869. Maximum Product of the Length of Two Palindromic Subsequences

[My Submissions \(/contest/weekly-contest-258/problems/maximum-product-of-the-length-of-two-palindromic-subsequences/submissions/\)](/contest/weekly-contest-258/problems/maximum-product-of-the-length-of-two-palindromic-subsequences/submissions/)
[Back to Contest \(/contest/weekly-contest-258/\)](/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:

subsequence1: e t e

s: l e e t c o d e c o m

subsequence2: c d c

Input: $s = \text{"leetcodecom"}$

Output: 9

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

Example 2:

Input: $s = \text{"bb"}$

Output: 1

Explanation: An optimal solution is to choose "b" (the first character) for the 1st subsequence and "b" (the second character) for the 2nd subsequence. The product of their lengths is: $1 * 1 = 1$.

Example 3:

Input: $s = \text{"accbcaxxcxx"}$

Output: 25

Explanation: An optimal solution is to choose "accba" for the 1st subsequence and "xxcxx" for the 2nd subsequence. The product of their lengths is: $5 * 5 = 25$.

Constraints:

- $2 \leq s.length \leq 12$
- s consists of lowercase English letters only.

JavaScript



```

1 const isPalindrome = (s) => { let n = s.length; let i = 0; let j = n - 1; while (i < j) { if (s[i++] !== s[j--]) return false; } return true; };
2 const counter = (a_or_s) => { let map = new Map(); for (const i of a_or_s) map.set(i, map.get(i) + 1 || 1); return map; };
3

```

```

4 ▼ 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++) {
15 ▼      if (i & (1 << j)) {
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) => y - x);
28  // pr(a);
29  let an = a.length;
30  let res = 0;
31 ▼  for (let i = 0; i < an; i++) {
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 ▼ const isDisjoint = (s, t, is, it) => {
44 ▼  for(const i of is) {
45    if (it.has(i)) return false;
46  }
47  return true;
48 };

```

☐ Custom Testcase☒ Use Example Testcases

Run

Submit

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 \(/region\)](#)