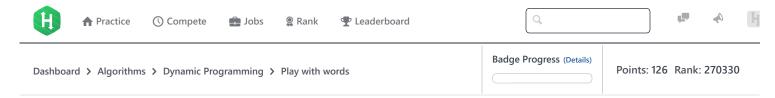
16/11/2017 HackerRank







Shaka and his brother have created a boring game which is played like this:

They take a word composed of lowercase English letters and try to get the maximum possible score by building exactly 2 **palindromic subsequences**. The score obtained is the product of the length of these 2 subsequences.

Let's say A and B are two subsequences from the initial string. If $A_i \otimes A_j$ are the smallest and the largest positions (from the initial word) respectively in A; and $B_i \otimes B_j$ are the smallest and the largest positions (from the initial word) respectively in B, then the following statements hold true:

 $A_i \leq A_j$

 $B_i \leq B_j$, &

 $A_j < B_i$

i.e., the positions of the subsequences should not cross over each other.

Hence the score obtained is the product of lengths of subsequences $A \otimes B$. Such subsequences can be numerous for a larger initial word, and hence it becomes harder to find out the maximum possible score. Can you help Shaka and his brother find this out?

Input Format

Input contains a word ${\it S}$ composed of lowercase English letters in a single line.

Constraints

 $1 < |S| \le 3000$

each character will be a lower case english alphabet.

Output Format

Output the maximum score the boys can get from S.

Sample Input

eeegeeksforskeeggeeks

Sample Output

50

Explanation

A possible optimal solution is eee-g-ee-ksfor-skeeggeeks being eeeee the one subsequence and skeeggeeks the other one. We can also select eegee in place of eeeee, as both have the same length.

f ⊌ in

 ${\color{red}\textbf{Submissions:}} \underline{1690}$

Max Score:65

16/11/2017 HackerRank

Difficulty: Medium
Rate This Challenge:
公公公公公

```
Current Buffer (saved locally, editable) & • •
                                                                                             Java 7
                                                                                                                               \Diamond
 1 ▼ import java.io.*;
 2 import java.util.*;
 3 import java.text.*;
 4 import java.math.*;
 5 import java.util.regex.*;
 7 ▼ public class Solution {
 8
        public static void main(String[] args) {
 9 ▼
            /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class should be named Solution. */
10 ▼
11
    }
12
                                                                                                                      Line: 1 Col: 1
                      ☐ Test against custom input
                                                                                                          Run Code
                                                                                                                       Submit Code
1 Upload Code as File
```

Join us on IRC at #hackerrank on freenode for hugs or bugs.

Contest Calendar | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature