All Contests > GCFL_3_year_6_sem > Remove Adjacents

Remove Adjacents

locked

Problem Submissions Leaderboard Discussions

You are given a string **s** consisting of lowercase Latin letters. Let the length of **s** be |**s**|. You may perform several operations on this string.

In one operation, you can choose some index **i** and remove the **i**-th character of s (s_i) if at least one of its adjacent characters is the previous letter in the Latin alphabet for s_i. For example, the previous letter for b is a, the previous letter for s is r, the letter a has no previous letters. Note that after each removal the length of the string decreases by one. So, the index **i** should satisfy the condition **1≤i≤|s|** during each operation.

For the character s_i adjacent characters are s_{i-1} and s_{i+1}. The first and the last characters of s both have only one adjacent character (unless |s|=1).

Consider the following example. Let s= bacabcab.

During the first move, you can remove the first character s_1= b because s_2= a. Then the string becomes s= acabcab. During the second move, you can remove the fifth character s_5= c because s_4= b. Then the string becomes s= acabab. During the third move, you can remove the sixth character s_6='b' because s_5= a. Then the string becomes s= acaba. During the fourth move, the only character you can remove is s_4= b, because s_3= a (or s_5= a). The string becomes s= acaa and you cannot do anything with it. Your task is to find the maximum possible number of characters you can remove if you choose the sequence of operations optimally

Input Format

The first line of the input contains one integer |s| (1≤|s|≤100) — the length of s.

The second line of the input contains one string s , consisting of |s| lowercase Latin letters

Constraints

1≤|s|≤100

Output Format

Print one integer — the maximum possible number of characters you can remove if you choose the sequence of moves optimally.

Sample Input 0

```
8
bacabcab
```

Sample Output 0

```
4
```

Sample Input 1

```
4
bcda
```

Sample Output 1

```
3
```

Explanation 1

In the second example, you can remove all but one character of s. The only possible answer follows.

During the first move, remove the third character s_3= d, s becomes bca. During the second move, remove the second character s_2= c, s becomes ba. And during the third move, remove the first character s_1= b, s becomes a.

Submissions: 0
Max Score: 0
Difficulty: Medium

Rate This Challenge:
☆☆☆☆☆

More

Current Buffer (saved locally, editable)

Java 8

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.*;

public class Main {
    static int mod=(int)1e9+7;
    static int max=0;
```

```java
    public static void main(String[] args) throws Exception {
        FastReader sc = new FastReader();
        int n=sc.nextInt();
        StringBuilder sb=new StringBuilder(sc.next());
        dfs(sb,0);
        System.out.println(max);
    }
    static void dfs(StringBuilder s,int m){
        if(s.length()==1)return;
        char c='a';
        int index=-1;
        for(int i=0;i<s.length();i++){
            if(i+1<s.length() && s.charAt(i)-1==s.charAt(i+1) && s.charAt(i)>c){
                index=i;
                c=s.charAt(i);
            }
            if(i-1>=0 && s.charAt(i)-1==s.charAt(i-1) && s.charAt(i)>c){
                index=i;
                c=s.charAt(i);
            }
        }
        for(int i=0;i<s.length();i++){
            if(i==index) {
                if (m + 1 > max) max = m + 1;
                StringBuilder temp = new StringBuilder(s.toString());
                dfs(temp.replace(i, i + 1, ""), m + 1);
            }
        }
    }
}
class FastReader {
    BufferedReader br;
    StringTokenizer st;

    public FastReader() {
        br = new BufferedReader(new
                InputStreamReader(System.in));
    }

    String next() {
        while (st == null || !st.hasMoreElements()) {
            try {
                st = new StringTokenizer(br.readLine());
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
        return st.nextToken();
    }

    int nextInt() {
        return Integer.parseInt(next());
    }

    long nextLong() {
        return Long.parseLong(next());
    }

    double nextDouble() {
        return Double.parseDouble(next());
    }

    String nextLine() {
        String str = "";
        try {
            str = br.readLine();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return str;
    }
}
```

Line: 1 Col: 1

Upload Code as File □ Test against custom input Run Code Submit Code

Testcase 0 ✓ Testcase 1 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
8
bacabcab
```

Your Output (stdout)

```
4
```

Expected Output

```
4
```

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