1454 Remove Palindromic Subsequences (link)

Description

You are given a string `s` consisting **only** of letters `'a'` and `'b'`. In a single step you can remove one **palindromic subsequence** from `s`.

Return *the **minimum** number of steps to make the given string empty*.

A string is a **subsequence** of a given string if it is generated by deleting some characters of a given string without changing its order. Note that a subsequence does **not** necessarily need to be contiguous.

A string is called **palindrome** if is one that reads the same backward as well as forward.

Example 1:

```
Input: s = "ababa"
Output: 1
Explanation: s is already a palindrome, so its entirety can be removed in a single ste
```

Example 2:

```
Input: s = "abb"
Output: 2
Explanation: "abb" -> "bb" -> "".
Remove palindromic subsequence "a" then "bb".
```

Example 3:

```
Input: s = "baabb"
Output: 2
Explanation: "baabb" -> "b" -> "".
Remove palindromic subsequence "baab" then "b".
```

Constraints:

- `1 <= s.length <= 1000`
- `s[i]` is either `'a'` or `'b'`.

(scroll down for solution)

Solution

Language: cpp

Status: Accepted

```cpp
#include <string>
using namespace std;

class Solution {
public:
    int removePalindromeSub(string s) {
        if (s.empty()) {
            return 0;
        }

        if (isPalindrome(s)) {
            return 1;
        } else {
            return 2;
        }
    }

private:
    bool isPalindrome(string& s) {
        int left = 0, right = s.length() - 1;
        while (left < right) {
            if (s[left] != s[right]) {
                return false;
            }
            left++;
            right--;
        }
        return true;
    }
};
```