680 Valid Palindrome II

2018年4月23日 17:17

Given a non-empty string s, you may delete **at most** one character. Judge whether you can make it a palindrome.

Example 1:

Input: "aba"

Output: True

Example 2:

Input: "abca"

Output: True

Explanation: You could delete the character 'c'.

Note:

1. The string will only contain lowercase characters a-z. The maximum length of the string is 50000.

来自 <https://leetcode.com/problems/valid-palindrome-ii/description/>

给定一个非空字符串 s，**最多**删除一个字符。判断是否能成为回文字符串。

注意:

1. 字符串只包含从 a-z 的小写字母。字符串的最大长度是50000。

Solution for Python3:

```python
class Solution1:
    def validPalindrome(self, s):
        """
        :type s: str
        :rtype: bool
        """
        def is_pali_range(i, j):
            return all(s[k] == s[j-k+i] for k in range(i, j))

        for i in range(len(s) // 2):
            if s[i] != s[~i]:
                j = len(s) - 1 -i
                return is_pali_range(i+1, j) or is_pali_range(i, j-1)
        return True

class Solution2:
    def validPalindrome(self, s):
        """
        :type s: str
        :rtype: bool
        """
        i = 0
        while i < len(s) // 2 and s[i] == s[~i]:
            i += 1
        s = s[i:len(s) - i]
        return s[1:] == s[1:][::-1] or s[:-1] == s[:-1][::-1]
```

Solution for C++:

```cpp
class Solution1 {
public:
    bool validPalindrome(string s) {
        int l = -1, r = s.length();
        while (++l < -- r)
            if (s[l] != s[r])
                return isPalindromic(s, l, r+1) || isPalindromic(s,
l-1, r);
        return true;
    }
    bool isPalindromic(string s, int l, int r) {
        while (++l < --r)
            if (s[l] != s[r])
                return false;
        return true;
    }
};

class Solution2 {
public:
    bool validPalindrome(string s) {
        for (int i = 0, j = s.length() - 1; i < j; i++, j--) {
            if (s[i] != s[j]) {
                int i1 = i, j1 = j - 1, i2 = i + 1, j2 = j;
                while (i1 < j1 && s[i1] == s[j1]) {
                    i1++;
                    j1--;
                }
                while (i2 < j2 && s[i2] == s[j2]) {
                    i2++;
                    j2--;
                }
                return i1 >= j1 || i2 >= j2;
            }
        }
        return true;
    }
};
```