Valid Palindrome II

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.

Solution 1

```
public boolean validPalindrome(String s) {
    int l = -1, r = s.length();
    while (++l < --r)
        if (s.charAt(l) != s.charAt(r)) return isPalindromic(s, l, r+1) || isPalindr
omic(s, l-1, r);
    return true;
}

public boolean isPalindromic(String s, int l, int r) {
    while (++l < --r)
        if (s.charAt(l) != s.charAt(r)) return false;
    return true;
}
```

written by compton_scatter original link here

Solution 2

Follow normal way (two pointers) to check if `s` is palindrome. When two chars are not equal, try to skip (pseudo `delete`) either left one or right one and continue checking.

```
class Solution {
    public boolean validPalindrome(String s) {
        int i = 0, j = s.length() - 1;
        while (i < j && s.charAt(i) == s.charAt(j)) {
            i++; j--;
        }

        if (i >= j) return true;

        if (isPalin(s, i + 1, j) || isPalin(s, i, j - 1)) return true;
        return false;
    }

    private boolean isPalin(String s, int i, int j) {
        while (i < j) {
            if (s.charAt(i) == s.charAt(j)) {
                i++; j--;
            }
            else return false;
        }
        return true;
    }
}
```

written by shawngao original link here

Solution 3

Compact

```
class Solution {
public:
    bool validPalindrome(string s) {
        return valid(s, 0, s.length() - 1, 1);
    }

private:
    bool valid(string& s, int i, int j, int d) { // d: num of chars you can delete a
t most
        return i >= j || (s[i] == s[j] ? valid(s, i + 1, j - 1, d) : d > 0 && (vali
d(s, i + 1, j, d - 1) || valid(s, i, j - 1, d - 1)));
    }
};
```

Cozy

```
class Solution {
public:
    bool validPalindrome(string s) {
        return valid(s, 0, s.length() - 1, 1);
    }

private:
    bool valid(string& s, int i, int j, int d) { // d: num of chars you can delete a
t most
        if (i >= j) return true;
        if (s[i] == s[j])
            return valid(s, i + 1, j - 1, d);
        else
            return d > 0 && (valid(s, i + 1, j, d - 1) || valid(s, i, j - 1, d - 1)
);
    }
};
```

written by alexander original link here