680. Valid Palindrome II

Easy 7.6K 385

Companies

Given a string `s`, return `true` *if the* `s` *can be palindrome after deleting* ***at most one*** *character from it.*

Example 1:

```
Input: s = "aba"
Output: true
```

Example 2:

```
Input: s = "abca"
Output: true
Explanation: You could delete the character 'c'.
```

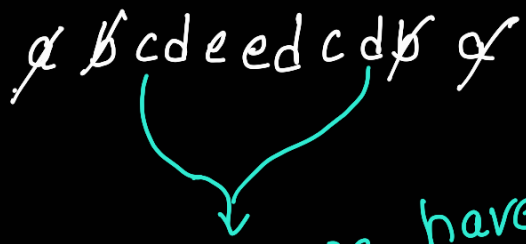
Example 3:

```
Input: s = "abc"
Output: false
```

Constraints:

- `1 <= s.length <= 10^5`
- `s` consists of lowercase English letters.

Accepted **624.5K** | Submissions **1.6M** | Acceptance Rate **39.5%**



here we have two options we could either delete 'c' or 'd' and move on but we can't really tell which removal is going to guarantee the valid palindrome. So we try both and if either of them makes valid palindrome then we return true or else false.

```
check (string s, int i, int j)
{
    while (i < j)
    {  if (s[i] != s[j])  return false
       else  i++, j--
    }
    return true
}

validPalindrome (String s)
{
    i = 0   j = n-1

    while (i < j)
```

```
{
    if (s[i] == s[j])  i++, j--
    else
        return check(s, i+1, j) // check(s, i, j-1)
}
if
    return true;
}
```

"aba" gets to this line.

here we deleted one character and now we are checking for valid palindrome without that character. If it comes out to be not a valid palindrome then it means we can't get a valid palindrome by removing one character. we don't care about removing more than 1 would make the valid palindrome or not.

$T(n) : O(n)$

$S(n) : O(1)$