409. Longest Palindrome

Dec. 27, 2017 | 47.3K views

Average Rating: 3.71 (31 votes)

Given a string which consists of lowercase or uppercase letters, find the length of the longest palindromes that can be built with those letters.

This is case sensitive, for example `"Aa"` is not considered a palindrome here.

Note:
Assume the length of given string will not exceed 1,010.

Example:

```
Input:
"abccccdd"

Output:
7

Explanation:
One longest palindrome that can be built is "dccaccd", whose length is 7.
```

Approach #1: Greedy [Accepted]

Intuition

A palindrome consists of letters with equal partners, plus possibly a unique center (without a partner). The letter `i` from the left has its partner `i` from the right. For example in `'abcba'`, `'aa'` and `'bb'` are partners, and `'c'` is a unique center.

Imagine we built our palindrome. It consists of as many partnered letters as possible, plus a unique center if possible. This motivates a greedy approach.

Algorithm

For each letter, say it occurs `v` times. We know we have `v // 2 * 2` letters that can be partnered for sure. For example, if we have `'aaaaa'`, then we could have `'aaaa'` partnered, which is `5 // 2 * 2 = 4` letters partnered.

At the end, if there was any `v % 2 == 1`, then that letter could have been a unique center. Otherwise, every letter was partnered. To perform this check, we will check for `v % 2 == 1` and `ans % 2 == 0`, the latter meaning we haven't yet added a unique center to the answer.

Java | Python — Copy

```python
class Solution:
    def longestPalindrome(self, s):
        ans = 0
        for v in collections.Counter(s).itervalues():
            ans += v / 2 * 2
            if ans % 2 == 0 and v % 2 == 1:
                ans += 1
        return ans
```

Complexity Analysis

- Time Complexity: $O(N)$, where N is the length of `s`. We need to count each letter.
- Space Complexity: $O(1)$, the space for our count, as the alphabet size of `s` is fixed. We should also consider that in a bit complexity model, technically we need $O(\log N)$ bits to store the count values.

Rate this article:

Previous Next

Comments: 22

Sort By

Type comment here... (Markdown is supported)

Preview Post

JPV ★ 726 March 7, 2019 5:46 PM

That's a lot of unneeded modulo operations. We don't need to construct the palindrome, or thus identify the specific center character, so we can just add 1 for a center character if needed at the very end. There is an extra character available anytime your answer is less than the length of the original string. E.g., after (OUTSIDE!) the loop, just do:

Read More

49 Share Reply

SHOW 2 REPLIES

Nevsanev ★ 1140 May 4, 2019 3:41 AM

Share my clean/concise Java solution:

```
class Solution {
    public int longestPalindrome(String s) {
        int[] count = new int[128];
```

Read More

22 Share Reply

SHOW 3 REPLIES

buerkuson ★ 44 January 31, 2019 12:25 PM

I have a solution using a hashset to check the final size of it and do a little modification.

```
    class Solution {
public int longestPalindrome(String s) {
    if (s == null || s.length() < 1) return 0;
```

Read More

19 Share Reply

Murgio ★ 19 December 5, 2018 12:55 AM

Python3 version:

```
def longestPalindrome(self, s):
    ans = 0
    for v in collections.Counter(s).values():
```

Read More

5 Share Reply

ngeek ★ 16 November 16, 2018 8:53 AM

Input sample is not clear for this question. We need not use all the characters of the given input.
Eg: aaaaabbb
Expected answer: 7 because output string is : aaaabbb
Not all characters of the letter 'a' needs to be considered in the final answer. which is what making this an easier question.

Read More

5 Share Reply

SHOW 1 REPLY

minalinskyy ★ 5 July 28, 2018 5:05 AM

The python code of answer missed a '/' in line 5. It should be ans += v // 2 * 2

5 Share Reply

SHOW 1 REPLY

zmono ★ 7 December 1, 2018 4:58 PM

`collections.Counter(s)` consumes O(N) space, as it has to read the whole string to tell you how many occurrences of each letters there are, so `.itervalues()` doesn't make it any better in this case.

1 Share Reply

sanoojag ★ 21 May 26, 2019 9:17 PM

We can make this more memory efficient.
int[] a=new int[52];

```
class Solution {
```

Read More

0 Share Reply

Serapqamar ★ 0 February 11, 2019 10:47 AM

There seems to be an error in the solution. Why would we increment answer for every odd length of a letter e.g. aaaaa would incrememnt answer by 1 and bbbbb would also increment the answer by 1. In that case we counted two unique centers which is not possible in a palindrome.

0 Share Reply

SHOW 3 REPLIES

sjw214 ★ 183 December 5, 2018 6:40 AM

Javascript solution:

```
var longestPalindrome = function(s) {
    const map = {}
    let pairs = 0;
```

Read More

0 Share Reply