Previous Next

125. Valid Palindrome

March 21, 2020 | 28.3K views

Average Rating: 3.91 (11 votes)

Given a string, determine if it is a palindrome, considering only alphanumeric characters and ignoring cases.

Note: For the purpose of this problem, we define empty string as valid palindrome.

Example 1:

```
Input: "A man, a plan, a canal: Panama"
Output: true
```

Example 2:

```
Input: "race a car"
Output: false
```

Constraints:

- `s` consists only of printable ASCII characters.

Solution

Approach 1: Compare with Reverse

Intuition

A palindrome is a word, phrase, or sequence that reads the same backwards as forwards. e.g. `madam`

A palindrome, and its reverse, are identical to each other.

Algorithm

We'll reverse the given string and compare it with the original. If those are equivalent, it's a palindrome.

Since only alphanumeric characters are considered, we'll filter out all other types of characters before we apply our algorithm.

Additionally, because we're treating letters as case-insensitive, we'll convert the remaining letters to lower case. The digits will be left the same.

C++ Java Python Copy

```
class Solution:
    def isPalindrome(self, s: str) -> bool:

        filtered_chars = filter(lambda ch: ch.isalnum(), s)
        lowercase_filtered_chars = map(lambda ch: ch.lower(), filtered_chars)

        filtered_chars_list = list(lowercase_filtered_chars)
        reversed_chars_list = filtered_chars_list[::-1]

        return filtered_chars_list == reversed_chars_list
```

Complexity Analysis

- Time complexity : $O(n)$, in length n of the string.

We need to iterate thrice through the string: 1. When we filter out non-alphanumeric characters, and convert the remaining characters to lower-case. 2. When we reverse the string. 3. When we compare the original and the reversed strings.

Each iteration runs linear in time (since each character operation completes in constant time). Thus, the effective run-time complexity is linear.

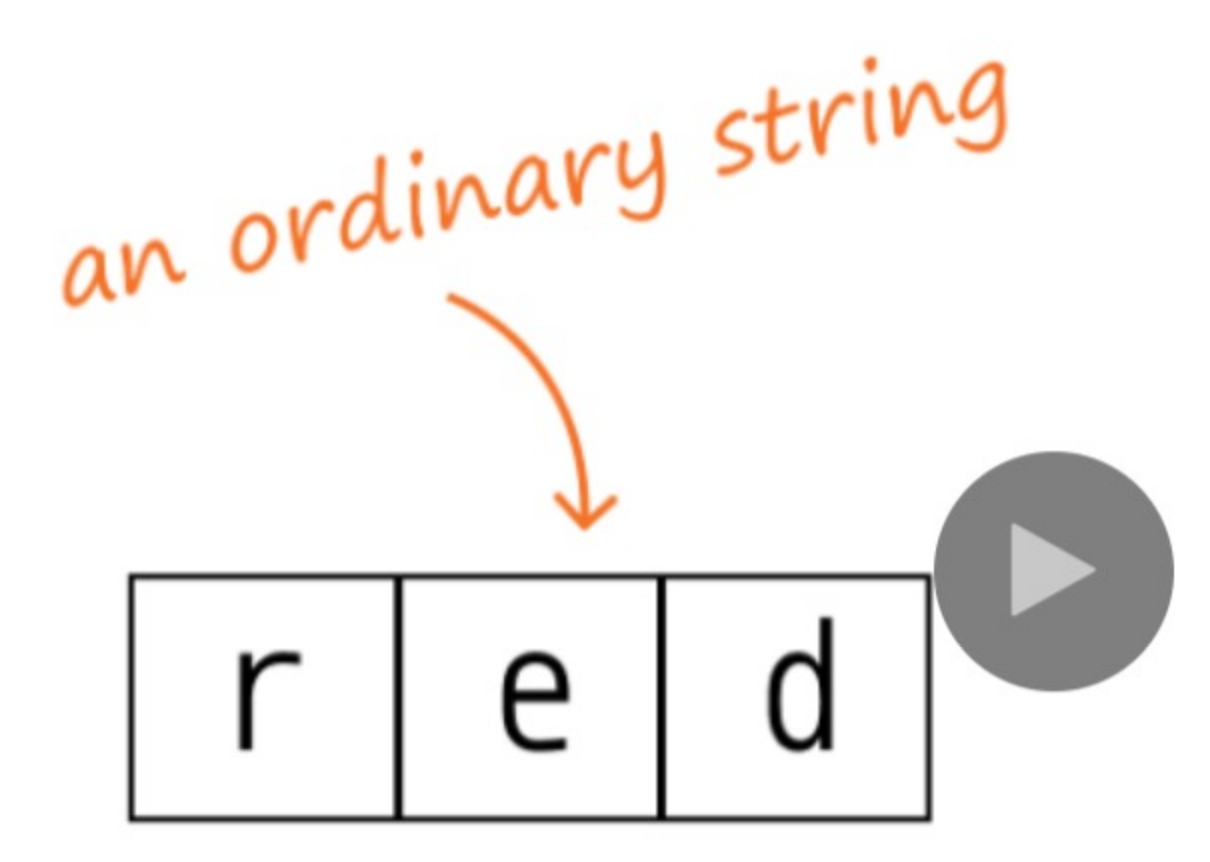
- Space complexity : $O(n)$, in length n of the string. We need $O(n)$ additional space to stored the filtered string and the reversed string.

Approach 2: Two Pointers

Intuition

If you take any ordinary string, and concatenate its reverse to it, you'll get a palindrome. This leads to an interesting insight about the converse: every palindrome half is reverse of the other half.

Simply speaking, if one were to start in the middle of a palindrome, and traverse outwards, they'd encounter the same characters, in the exact same order, in both halves!



1 / 6

Algorithm

Since the input string contains characters that we need to ignore in our palindromic check, it becomes tedious to figure out the real middle point of our palindromic input.

> Instead of going outwards from the middle, we could just go inwards towards the middle!

So, if we start traversing inwards, from both ends of the input string, we can expect to *see* the same characters, in the same order.

The resulting algorithm is simple: + Set two pointers, one at each end of the input string + If the input is palindromic, both the pointers should point to equivalent characters, *at all times*. [1] + If this condition is not met at any point of time, we break and return early. [2] + We can simply ignore non-alphanumeric characters by continuing to traverse further. + Continue traversing inwards until the pointers meet in the middle.

C++ Java Python Copy

```
1  class Solution:
2      def isPalindrome(self, s: str) -> bool:
3
4          i, j = 0, len(s) - 1
5
6          while i < j:
7              while i < j and not s[i].isalnum():
8                  i += 1
9              while i < j and not s[j].isalnum():
10                 j -= 1
11
12             if i < j and s[i].lower() != s[j].lower():
13                 return False
14
15             i += 1
16             j -= 1
17
18         return True
19
```

Complexity Analysis

- Time complexity : $O(n)$, in length n of the string. We traverse over each character at-most once, until the two pointers meet in the middle, or when we break and return early.
- Space complexity : $O(1)$. No extra space required, at all.

1. Such a property is formally known as a loop invariant. ↩
2. Such a property is often called a *loop termination condition*. It is one of several used in this solution. Can you identify the others? ↩

Rate this article:

Previous Next

Comments: 10 Sort By

Type comment here... (Markdown is supported)

Preview Post

David_E_78 ★23 June 25, 2020 4:51 AM

This test case, using the double pointer solution, is failing on my Mac OSX Mojave
"abb'a"
Looks like some character encoding issue.
Anyone else seeing the same?

11 Share Reply

SHOW 8 REPLIES

joaoh82 ★10 May 26, 2020 6:07 PM

I rather use left and right variables names then i and j. This way more readable.

```
def isPalindrome(self, s: str) -> bool:
    left, right = 0, len(s)-1
```

Read More

4 Share Reply

pimtchenkov ★2 April 22, 2020 8:17 AM

Description doesn't say that string should have Engl. letters only! And test cases interpret input "ϕϕϕ шшш" as true. Either one should be fixed.

2 Share Reply

SHOW 2 REPLIES

RobMaldo ★1 May 25, 2020 8:15 PM

As a way to see it clear:

```
def isPalindrome(self, s: str) -> bool:
  s = "".join([c.lower() for c in s if c not in string.punctuation and c != " "])
```

Read More

1 Share Reply

vedanshleetcode2 ★6 April 5, 2020 7:42 AM

In the last solution,
I think we don't have to check " if i < j " in line 12.
We can just check " s[i].lower() != s[j].lower() "

1 Share Reply

SHOW 1 REPLY

lordhyde ★0 2 days ago

php easy solution using regular expressions

```
    function isPalindrome($s) {
        $s = preg_replace( '/[^a-z0-9]/', '', strtolower($s));
```

Read More

0 Share Reply

pratyakshrana ★0 July 6, 2020 10:19 AM

What If i use HashMap in place of StringBuilder + reverse method + equals method: Below is my code in C#.

```
class Solution {
```

Read More

0 Share Reply

CELIANG ★0 June 27, 2020 2:10 AM

My python code got "abb'a" -> True on my own pc, but the result returned from Leetcode showed False, and I got 481/482 tests passed, except for this one. Anyone else got the same problem?

0 Share Reply

SHOW 2 REPLIES

WilmerKrisp ★15 June 26, 2020 3:27 PM

terrible! Approach 2: Two Pointers DOESNOT WORK!!
bad quality leetcode material:(((((

Testcase: "abb'a"

Read More

0 Share Reply

SHOW 2 REPLIES

mukkact ★0 June 24, 2020 7:36 AM

Could be done with one loop

class Solution {
public boolean isPalindrome(String s) {

Read More

0 Share Reply

SHOW 1 REPLY