9. Palindrome Number

Aug. 31, 2017 | 352.9K views

Previous Next

Average Rating: 4.29 (248 votes)

Determine whether an integer is a palindrome. An integer is a palindrome when it reads the same backward as forward.

Example 1:

```
Input: 121
Output: true
```

Example 2:

```
Input: -121
Output: false
Explanation: From left to right, it reads -121. From right to left, it becomes 121-. T
```

Example 3:

```
Input: 10
Output: false
Explanation: Reads 01 from right to left. Therefore it is not a palindrome.
```

Follow up:

Coud you solve it without converting the integer to a string?

Solution

Approach 1: Revert half of the number

Intuition

The first idea that comes to mind is to convert the number into string, and check if the string is a palindrome, but this would require extra non-constant space for creating the string which is not allowed by the problem description.

Second idea would be reverting the number itself, and then compare the number with original number, if they are the same, then the number is a palindrome. However, if the reversed number is larger than int.MAX, we will hit integer overflow problem.

Following the thoughts based on the second idea, to avoid the overflow issue of the reverted number, what if we only revert half of the int number? After all, the reverse of the last half of the palindrome should be the same as the first half of the number, if the number is a palindrome.

For example, if the input is `1221`, if we can revert the last part of the number "12**21**" from "**21**" to "**12**", and compare it with the first half of the number "12", since 12 is the same as 12, we know that the number is a palindrome.

Let's see how we could translate this idea into an algorithm.

Algorithm

First of all we should take care of some edge cases. All negative numbers are not palindrome, for example: -123 is not a palindrome since the '-' does not equal to '3'. So we can return false for all negative numbers.

Now let's think about how to revert the last half of the number. For number `1221`, if we do `1221 % 10`, we get the last digit `1`, to get the second to the last digit, we need to remove the last digit from `1221`, we could do so by dividing it by 10, `1221 / 10 = 122`. Then we can get the last digit again by doing a modulus by 10, `122 % 10 = 2`, and if we multiply the last digit by 10 and add the second last digit, `1 * 10 + 2 = 12`, it gives us the reverted number we want. Continuing this process would give us the reverted number with more digits.

Now the question is, how do we know that we've reached the half of the number?

Since we divided the number by 10, and multiplied the reversed number by 10, when the original number is less than the reversed number, it means we've processed half of the number digits.

C# Copy

```
    public bool IsPalindrome(int x) {
        // Special cases:
        // As discussed above, when x < 0, x is not a palindrome.
        // Also if the last digit of the number is 0, in order to be a palindrome,
        // the first digit of the number also needs to be 0.
        // Only 0 satisfy this property.
        if(x < 0 || (x % 10 == 0 && x != 0)) {
            return false;
        }

        int revertedNumber = 0;
        while(x > revertedNumber) {
            revertedNumber = revertedNumber * 10 + x % 10;
            x /= 10;
        }

        // When the length is an odd number, we can get rid of the middle digit by revertedNumber/10
        // For example when the input is 12321, at the end of the while loop we get x = 12, revertedNumber = 123,
        // since the middle digit doesn't matter in palidrome(it will always equal to itself), we can simply get rid of it.
        return x == revertedNumber || x == revertedNumber/10;
    }
}
```

Complexity Analysis

- Time complexity : $O(\log_{10}(n))$. We divided the input by 10 for every iteration, so the time complexity is $O(\log_{10}(n))$
- Space complexity : $O(1)$.

Rate this article:

Previous Next

Comments: 500

Sort By

Type comment here... (Markdown is supported)

Preview Post

fortuna911 ★935 March 14, 2019 4:21 PM

"but this would require extra non-constant space for creating the string which is not allowed by the problem description."

Problem description doesn't say we must use constant space.

217 Share Reply

SHOW 6 REPLIES

athiyadeviyani ★217 December 20, 2018 12:47 AM

Python 3 one-liner
def isPalindrome(self, x):
return str(x) == str(x)[::-1]

216 Share Reply

SHOW 19 REPLIES

anthonybusto22 ★100 October 9, 2018 12:59 PM

Here is a clean Java solution, without converting the input number to a string.

```
public boolean isPalindrome(int num){
    if(num < 0) return  false;
```

Read More

81 Share Reply

SHOW 10 REPLIES

lengyanhua ★97 July 26, 2018 6:48 AM

```
public boolean isPalindrome(int x) {
    String str = String.valueOf(x);
    int start = 0;
    int end = str.length() - 1;
```

Read More

97 Share Reply

SHOW 5 REPLIES

JamesCool ★77 February 18, 2019 3:23 AM

return str(x) == str(x)[::-1]
为所欲为.jpeg

61 Share Reply

SHOW 7 REPLIES

ZheYuan ★45 August 22, 2018 7:51 PM

python:

```
return False if x < 0 else x == int(str(x)[::-1])
```

45 Share Reply

SHOW 7 REPLIES

fanyank ★24 October 13, 2018 7:53 AM

```
if(x == 0) {
            return true;
        }
        if(x < 0 || x%10 == 0)
```

Read More

24 Share Reply

SHOW 1 REPLY

zhengzhicong ★294 November 7, 2018 6:56 PM Report

python3:

```
class Solution:
    def isPalindrome(self, x):
        """
```

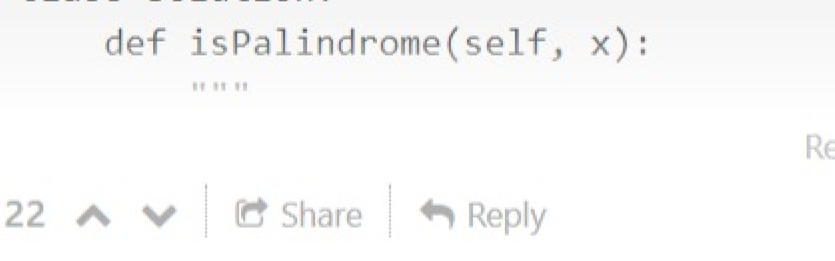
Read More

22 Share Reply

SHOW 2 REPLIES

terrible_whiteboard ★626 May 19, 2020 7:26 AM

I made a video if anyone is having trouble understanding the solution (clickable link)
https://youtu.be/gnwFjlUXN1o



Read More

21 Share Reply

taoshi ★32 July 27, 2018 1:29 PM Report

My python3 solution
class Solution:
def isPalindrome(self, x):
""""
:type x: int

Read More

19 Share Reply

SHOW 2 REPLIES

‹ 1 2 3 4 5 6 ... 49 50 ›