链滴

算法 (3) 回文数

作者：ws

原文链接：https://ld246.com/article/1541219672806

来源网站：链滴

判断一个整数是否是回文数。回文数是指正序（从左向右）和倒序（从右向左）读都是一样的整数。

示例 1:

输入: 121
输出: true

示例 2:

输入: -121
输出: false
解释: 从左向右读, 为 -121 。 从右向左读, 为 121- 。因此它不是一个回文数。

示例 3:

输入: 10
输出: false
解释: 从右向左读, 为 01 。因此它不是一个回文数。

进阶:

你能不将整数转为字符串来解决这个问题吗?

答案

反转比较

```
class Solution {
    public static boolean isPalindrome(int x) {
        if(x < 0){
            return false;
        }
        int result =0;
        int temp = x;
        while(temp !=0){
            result = temp%10 + result*10;
            temp =temp/10;
        }
        if(x == result){
            return true;
        }
        return false;
    }
}
```

折半比较

```
public class Solution {
    public bool IsPalindrome(int x) {
        // 特殊情况：
        // 如上所述，当 x < 0 时，x 不是回文数。
        // 同样地，如果数字的最后一位是 0，为了使该数字为回文，
        // 则其第一位数字也应该是 0
```

```
        // 只有 0 满足这一属性
        if(x < 0 || (x % 10 == 0 && x != 0)) {
            return false;
        }

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

        // 当数字长度为奇数时，我们可以通过 revertedNumber/10 去除处于中位的数字。
        // 例如，当输入为 12321 时，在 while 循环的末尾我们可以得到 x = 12，revertedNumber =
23，
        // 由于处于中位的数字不影响回文（它总是与自己相等），所以我们可以简单地将其去除。
        return x == revertedNumber || x == revertedNumber/10;
    }
}
```