1. 题目
2. 题干

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。

1. 示例

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。

1. 思路
2. 数字转string字符串比较

首先想到的都是转字符串，然后首尾连续比较，这种方法非常简单，速度也挺快。将数字转为字符串，从字符串的首字符开始，相对于尾部开始的字符串，一一比较，如有不同，则返回false。

1. 巧妙思路：边取尾数边比较

利用数学方法取尾数，利用每次取的尾数来构造整数reverse，因为我们每次得到的尾数的顺序和前面数字的顺序是一样的，通过比较尾数构造的整数reverse和剩余整数x的大小，就可以得知我们是否已经操作到了原始数字尾数的一半了。

如果reverse >=x，则说明已经操作到了原始数字位数的一半了。

当原数位数为偶数且是回文数时，则结果x = reverse；

当袁术位数为奇数且是回文串时，则结果x = reverse / 10;

其他结果均为false

注意：有一些必定为false的回文数一开始就直接排除掉：

X < 0;

X % 10 ==0 && x !+ 0

1. 代码实现
2. string实现

class Solution {

public:

bool isPalindrome(int x) {

//用string解决

string s = to\_string(x);

bool flag = true;

for (int i = 0; i < s.length()/ 2; ++i) {

if (s[i] != s[s.length() - i]) {

flag = false;

break;

}

}

return flag;

}

};

1. 弹出尾数

class Solution {

public:

bool isPalindrome(int x) {

//两种一定false

if (x < 0 || (x % 10 == 0 && x != 0))

return false;

int reverse = 0;

while (x > reverse) {

reverse = reverse \* 10 + x % 10;

x /= 10;

}

return x == reverse || x == reverse / 10;

}

};