# 9、回文数（palindrome-number）

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？

## 思路：

* 如果我们使用字符串来解答：

static void Main(string[] args)  
{  
 int x = 1221;  
 bool b = IsPalindrome1(x);  
 Console.WriteLine(b);  
 Console.ReadKey();  
}  
  
//简单做法 使用字符串  
public static bool IsPalindrome1(int x)  
{  
 if (x < 0)  
 {  
 return false;  
 }  
 var arr = x.ToString().ToCharArray();  
 Array.Reverse(arr);  
 return x.ToString() == new string(arr);  
}

* 使用像第七题那样的逆序输出最后比较

public static bool IsPalindrome(int x)  
{  
 //当这个数是0或者负数都不存在  
 //(x % 10 == 0 && x != 0)  
 //这里值得注意的是我们使用的反转方法是和7的做法一样，但是这种方法无法判断10这样的数字  
 //还可以进行优化  
 if (x < 0 || (x % 10 == 0 && x != 0))  
 {  
 return false;  
 }  
 //利用回文的性质，先计算后一半的逆序值是否和前一半相等  
 //即可判断  
 int res = 0;  
 //不断的除 不断变小当x小于或者等于res时候 说明x被分成两半了  
 while (x > res)  
 {  
 res = res \* 10 + x % 10;  
 x /= 10;  
 }  
 //res == x表示偶数，例如1221时21被转成12就和12相等  
 //而121表示1，之所以判断加在x /= 10之前 此时res还没变化 反转的12和源数字的12相等  
 return res / 10 == x || res == x;  
}