**2. Add Two Numbers**

Medium

85332163Add to ListShare

You are given two **non-empty** linked lists representing two non-negative integers. The digits are stored in **reverse order** and each of their nodes contain a single digit. Add the two numbers and return it as a linked list.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

**Example:**

**Input:** (2 -> 4 -> 3) + (5 -> 6 -> 4)

**Output:** 7 -> 0 -> 8

**Explanation:** 342 + 465 = 807.

题目，243就是342， 564代表着465

让这两个数加起来

=807

再reverse

注意，每个Node都代表着digit，因此不会有两位数这种，只有一位数

思路：

其实这就是个数学问题

2 4 3

+ 5 6 4=

7 0 8 //进位在右边

如果是243+ 5641这种，短的那个填充0

2 4 3 0

5 6 4 1=

7 0 8 1

/\*\*

\* Definition for singly-linked list.

\* public class ListNode {

\* int val;

\* ListNode next;

\* ListNode() {}

\* ListNode(int val) { this.val = val; }

\* ListNode(int val, ListNode next) { this.val = val; this.next = next; }

\* }

\*/

class Solution {

public ListNode addTwoNumbers(ListNode l1, ListNode l2) {

ListNode l3=new ListNode(0);

ListNode head=l3; //head作为备用，因为一会儿l3会不停next，我们需要一个head来return, 常见手段

int carry=0; //进位，一开始是0

while (l1!=null||l2!=null){ 除非两个都是Null代表结束，不然一直继续

int temp1=(l1!=null)?l1.val:0; //等于null代表他是短的，填充0

int temp2=(l2!=null)?l2.val:0;

int digit=(temp1+temp2+carry)%10; 个位

carry=(temp1+temp2+carry-digit)/10; 进位

l3.next=new ListNode(digit); next=个位

l3=l3.next;

if(l1!=null) l1=l1.next;

if(l2!=null) l2=l2.next;

}

if (carry>0)

l3.next=new ListNode(carry); 在全部结束以后，carry理论上会更新，如果还有carry，那就最后额外加一下

return head.next; //注意head是我们随便填的0，真正的head是他的next

}

}