两数相加

给出两个 非空 的链表用来表示两个非负的整数。其中，它们各自的位数是按照 逆序 的方式存储的，并且它们的每个节点只能存储 一位 数字。

如果，我们将这两个数相加起来，则会返回一个新的链表来表示它们的和。

您可以假设除了数字 0 之外，这两个数都不会以 0 开头。

示例：

输入：(2 -> 4 -> 3) + (5 -> 6 -> 4)

输出：7 -> 0 -> 8

原因：342 + 465 = 807

解题思路：

将链表 1 和链表 2 对应节点相加，我们知道相加会产生进位，进位需要标志位来操作

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {
        int len1=1;//记录 l1 的长度
        int len2=1;//记录 l2 的长度
        ListNode* p=l1;
        ListNode* q=l2;
        while(p->next!=NULL)//获取 l1 的长度
        {
            len1++;
            p=p->next;
        }
        while(q->next!=NULL)//获取 l2 的长度
        {
            len2++;
            q=q->next;
        }
        if(len1>len2)//l1 较长，在 l2 末尾补零
```

```
        {
            for(int i=1;i<=len1-len2;i++)
            {
                q->next=new ListNode(0);
                q=q->next;
            }
        }
        else//l2 较长，在 l1 末尾补零
        {
            for(int i=1;i<=len2-len1;i++)
            {
                p->next=new ListNode(0);
                p=p->next;
            }
        }
        p=l1;
        q=l2;
        bool count=false;//记录进位
        ListNode* l3=new ListNode(-1);//存放结果的链表
        ListNode* w=l3;//l3 的移动指针
        int i=0;//记录相加结果
        while(p!=NULL&&q!=NULL)
        {
            i=count+p->val+q->val;
            w->next=new ListNode(i%10);
            count=i>=10?true:false;
            w=w->next;
            p=p->next;
            q=q->next;
        }
        if(count)//若最后还有进位
        {
            w->next=new ListNode(1);
            w=w->next;
        }
        return l3->next;
    }
};
```