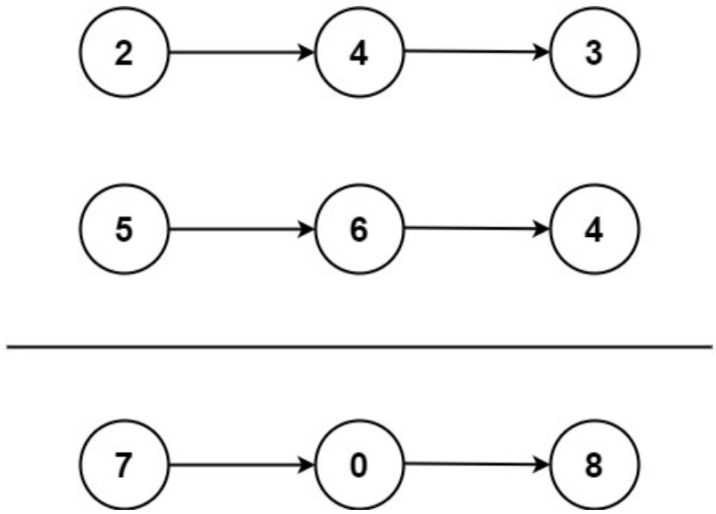
2.两数相加

给你两个 **非空** 的链表，表示两个非负的整数。它们每位数字都是按照 **逆序** 的方式存储的，并且每个节点只能存储 **一位** 数字。

请你将两个数相加，并以相同形式返回一个表示和的链表。

你可以假设除了数字 0 之外，这两个数都不会以 0 开头。

示例 1：



> **输入：** l1 = [2,4,3], l2 = [5,6,4]
> **输出：** [7,0,8]
> **解释：** 342 + 465 = 807.

示例 2：

> **输入：** l1 = [0], l2 = [0]
> **输出：** [0]

示例 3：

> **输入：** l1 = [9,9,9,9,9,9,9], l2 = [9,9,9,9]
> **输出：** [8,9,9,9,0,0,0,1]

提示：

- 每个链表中的节点数在范围 `[1, 100]` 内
- `0 <= Node.val <= 9`
- 题目数据保证列表表示的数字不含前导零

题目将数字**逆序存放**，所以可以直接从两个链表的头开始对应元素相加，并使用一个变量记录进位。

链表操作，可以设置一个预取节点`preNode`用于指向第一个节点，后续操作更方便

Code：

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
```

```cpp
 *     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) {
        ListNode *preNode = new ListNode(-1);
        ListNode *nowNode = preNode;
        bool flag=false;
        while(true){
            // 处理逐元素相加
            if(l1==NULL&&l2==NULL) break;
            int newNumber=(l1==NULL?0:l1->val)+(l2==NULL?0:l2->val)+(flag);
            flag = newNumber>=10;
            nowNode->next = new ListNode(newNumber%10);
            nowNode = nowNode->next;
            if(l1!=NULL) l1 = l1->next;
            if(l2!=NULL) l2 = l2->next;
        }
        if(flag) nowNode->next = new ListNode(1);

        return preNode->next;
    }
};
```