- https://leetcode-cn.com/problems/add-two-numbers/

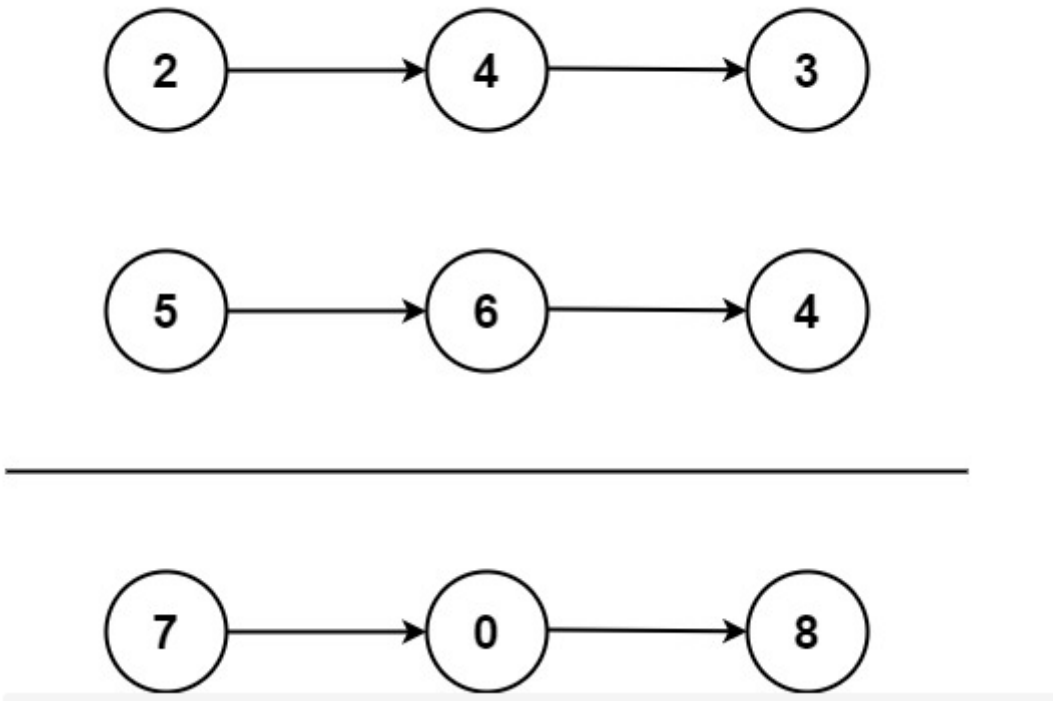
两数相加

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：



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

解法一

从左到右依次遍历链表，对相同位置的数字依次相加，同时定义一个进位值 carry，则对于 0 位置它的值为 `(n1 + n2 + carry) % 10`，新的 carry 值为 `(n1 + n2 + carry) / 10`,对于较短的链表，其后面的位置默认赋值为 0，最后如果 carry > 0,则需要在链表末尾加入一个节点 carry。

```
/**
 * 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 head = null;
        ListNode tail = null;
        int carry = 0;
        while(l1 != null || l2 != null) {
            int n1 = l1 != null ? l1.val : 0;
            int n2 = l2 != null ? l2.val : 0;

            int sum = (n1 + n2 + carry) % 10;
            carry = (n1 + n2 + carry) / 10;
            if (head == null) {
                head = tail = new ListNode(sum);
            } else {
                tail.next = new ListNode(sum);
                tail = tail.next;
            }
            if (l1 != null) {
                l1 = l1.next;
            }

            if (l2 != null) {
                l2 = l2.next;
            }
        }
```

```
        if (carry > 0) {
                tail.next = new ListNode(carry);
                tail = tail.next;
        }
        return head;
    }
}
```

复杂度分析

- 时间复杂度：`O(max(m,n))`，其中 m,n 为两个链表的长度。我们要遍历两个链表的全部位置，而处理每个位置只需要 `O(1)` 的时间。
- 空间复杂度：`O(max(m,n))`。答案链表的长度最多为较长链表的长度 +1。