一、题目要求

题目2. Add Two Numbers，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.
```

二、我的解答

由于english比较low，理解上述题目还是花了点时间。

题目看懂了，确实不难，涉及结构体、指针，求和。

然后就开工，直接在线写代码，编译通过，但是提交后报错了：

1.第一次错误是Runtime Error，具体错误是

signed integer overflow: 1000000000000000000 * 10 cannot be represented in

2.第二次错误**AddressSanitizer: heap-use-after-free on address 0x602000000118 at pc 0x000000462f75 bp 0x7fff9680bfd0 sp 0x7fff9680bfc8**

后来仔细考虑了一下，我做的过程是：

将链表转换为一个整数（用了long long），然后求和，最后转换为一个链表返回。

这个题目，我考虑复杂了。错误之处在于链表表示的数，可以非常大，也可以是0。

下面是我的错误代码：

```
class Solution {
public:
    ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {
        long long n1 = 0;
        long long n2 = 0;
        long long result = 0;
        long long t = 1;
        ListNode *p = l1;
        while(p != NULL){
            n1 = n1 + t* p->val;
            t = t* 10;
            p = p->next;
        }

        p = l2;
        t = 1;
        while(p != NULL){
            n2 = n2 + t* p->val;
            t = t * 10;
```

```
            p = p->next;
        }
        result = n1 + n2;

        ListNode * pHead = NULL;
        if(result == 0){
            return pHead = new ListNode(0);
        }
        while(result>0){
            if(pHead == NULL){
                pHead = new ListNode(result % 10);
            }else{
                p = pHead;
                while(p->next !=NULL){
                    p = p ->next;
                }
                p->next = new ListNode(result % 10);
            }

            result = result / 10;
        }
        return pHead;
    }
};
```

直接链表对应为求和，然后返回链表就可以了，这个反而更简单。完整的代码如下：

```
#include<iostream>
using namespace std;

struct ListNode {
    int val;
    ListNode *next;
    ListNode(int x) : val(x), next(NULL) {}
};

class Solution {
public:
    ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {
        ListNode head(0),*curr = & head;
        int remain=0,tmp;
        while(l1!=NULL && l2!=NULL){
            tmp = l1->val + l2->val + remain;
            curr -> next = new ListNode(tmp % 10);
            curr = curr->next;
            l1 = l1->next;
            l2 = l2->next;
            remain = tmp / 10;
        }
        while(l1 !=NULL){
            tmp = l1->val + remain;
            curr->next = new ListNode(tmp % 10);
            curr = curr->next;
            l1 = l1->next;
            remain = tmp / 10;
```

```
        }
        while(l2 !=NULL){
            tmp = l2->val + remain;
            curr->next= new ListNode(tmp % 10);
            curr = curr->next;
            l2 = l2->next;
            remain = tmp /10;
        }
        if(remain !=NULL){
            curr->next = new ListNode(remain);
        }
        return head.next;
    }
};

int main(){
    Solution s;
    ListNode * l1,*l2,*curr;

    //初始化 l1 2->4->3
    l1= new ListNode(2);
    curr = l1;
    curr->next = new ListNode(4);
    curr = curr->next;
    curr->next = new ListNode(3);
    curr = curr->next;

    //初始化 l2 5->6->4
    l2= new ListNode(5);
    curr = l2;
    curr->next = new ListNode(6);
    curr = curr->next;
    curr->next = new ListNode(4);
    curr = curr->next;

    ListNode * l3 = s.addTwoNumbers(l1,l2);

    //输出结果
    curr = l3;
    while(curr!=NULL){
        cout<<curr->val<<" ";
        curr= curr->next;
    }

    return 0;
}
```