Day – 53 of the #101 days of the coding challenge-----------

Problem:- 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 contains a single digit. Add the two numbers and return the sum as a linked list.

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

Example:- **Input:** l1 = [2,4,3], l2 = [5,6,4]
Output: [7,0,8]
Explanation: 342 + 465 = 807.

Example 2:

Input: l1 = [0], l2 = [0]
Output: [0]

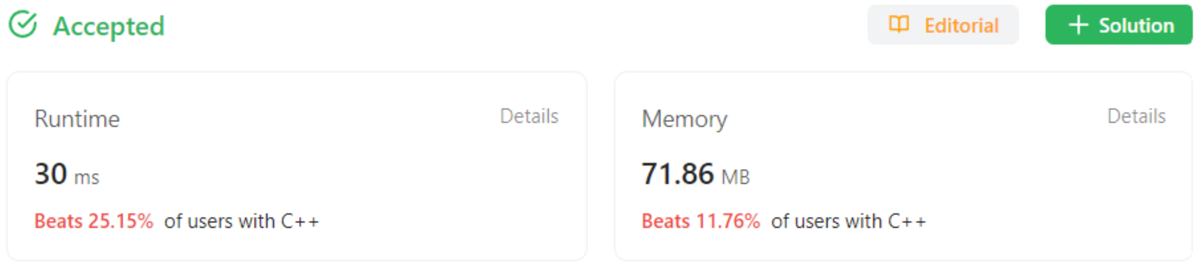
Example 3:

Input: l1 = [9,9,9,9,9,9,9], l2 = [9,9,9,9]
Output: [8,9,9,9,0,0,0,1]

Code:-

```
class Solution {

public:

   ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {



   // creating new ListNode for storing the addition value

   ListNode * l3 = new ListNode(0); // initilizing with zero



   int carry = 0;
}

ListNode *head = l3;
// till l1 and l2 is having the value
while(l1 && l2){

    int value = l1->val + l2->val+carry;
    carry = value/10;
    l3-> next = new ListNode(value % 10);
    l3 = l3->next;
    l1 = l1->next;
    l2 = l2->next;
}

while(l1)
{
    int value = l1->val + carry;
    carry = value/10;
    l3-> next = new ListNode(value % 10);
    l3 = l3->next;
    l1 = l1->next;
}
```

```
    while(l2)
    {
        int value = l2->val + carry;
        carry = value/10;
        l3-> next = new ListNode(value%10);
        l3 = l3->next;
        l2 = l2->next;
    }

    if(carry)
    {
        l3->next = new ListNode(carry);
    }

    return head->next;

     }
};
```



Accepted

Runtime

30 ms

Beats 25.15% of users with C++

Memory

71.86 MB

Beats 11.76% of users with C++