Add Two Numbers

You are given two linked lists representing two non-negative numbers. 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.

```
Input: (2 -> 4 -> 3) + (5 -> 6 -> 4)
Output: 7 -> 0 -> 8
```

Solution 1

```java
public class Solution {
    public ListNode addTwoNumbers(ListNode l1, ListNode l2) {
        ListNode c1 = l1;
        ListNode c2 = l2;
        ListNode sentinel = new ListNode(0);
        ListNode d = sentinel;
        int sum = 0;
        while (c1 != null || c2 != null) {
            sum /= 10;
            if (c1 != null) {
                sum += c1.val;
                c1 = c1.next;
            }
            if (c2 != null) {
                sum += c2.val;
                c2 = c2.next;
            }
            d.next = new ListNode(sum % 10);
            d = d.next;
        }
        if (sum / 10 == 1)
            d.next = new ListNode(1);
        return sentinel.next;
    }
}
```

written by potpie original link here

Solution 2

```
ListNode *addTwoNumbers(ListNode *l1, ListNode *l2) {
    ListNode preHead(0), *p = &preHead;
    int extra = 0;
    while (l1 || l2 || extra) {
        int sum = (l1 ? l1->val : 0) + (l2 ? l2->val : 0) + extra;
        extra = sum / 10;
        p->next = new ListNode(sum % 10);
        p = p->next;
        l1 = l1 ? l1->next : l1;
        l2 = l2 ? l2->next : l2;
    }
    return preHead.next;
}
```

written by ce2 original link here

Solution 3

Two things to make the code simple:

1. Whenever one of the two *ListNode* is null, replace it with 0.
2. Keep the while loop going when at least one of the three conditions is met.

Let me know if there is something wrong. Thanks.

```
public class Solution {
    public ListNode addTwoNumbers(ListNode l1, ListNode l2) {
        ListNode prev = new ListNode(0);
        ListNode head = prev;
        int carry = 0;
        while (l1 != null || l2 != null || carry != 0) {
            ListNode cur = new ListNode(0);
            int sum = ((l2 == null) ? 0 : l2.val) + ((l1 == null) ? 0 : l1.val) +
carry;
            cur.val = sum % 10;
            carry = sum / 10;
            prev.next = cur;
            prev = cur;

            l1 = (l1 == null) ? l1 : l1.next;
            l2 = (l2 == null) ? l2 : l2.next;
        }
        return head.next;
    }
}
```

written by dchen0215 original link here