Plus One Linked List

Given a non-negative number represented as a singly linked list of digits, plus one to the number.

The digits are stored such that the most significant digit is at the head of the list.

Example:

```
Input:
1->2->3

Output:
1->2->4
```

Solution 1

```
public class Solution {
    public ListNode plusOne(ListNode head) {
        ListNode dummy = new ListNode(0);
        dummy.next = head;
        ListNode i = dummy;
        ListNode j = dummy;

        while (j.next != null) {
            j = j.next;
            if (j.val != 9) {
                i = j;
            }
        }

        if (j.val != 9) {
            j.val++;
        } else {
            i.val++;
            i = i.next;
            while (i != null) {
                i.val = 0;
                i = i.next;
            }
        }

        if (dummy.val == 0) {
            return dummy.next;
        }

        return dummy;
    }
}
```

- i stands for the most significant digit that is going to be incremented if there exists a carry
- dummy node can handle cases such as "9->9>-9" automatically

written by ocean original link here

Solution 2

At the first glance, I want to reverse the inputs, add one, then reverse back. But that is too intuitive and I don't think this is an expected solution. Then what kind of alg would adding one in reverse way for list?

Recursion! With recursion, we can visit list in reverse way! So here is my recursive solution.

```
public ListNode plusOne(ListNode head) {
    if( DFS(head) == 0){
        return head;
    }else{
        ListNode newHead = new ListNode(1);
        newHead.next = head;
        return newHead;
    }
}

public int DFS(ListNode head){
    if(head == null) return 1;

    int carry = DFS(head.next);

    if(carry == 0) return 0;

    int val = head.val + 1;
    head.val = val%10;
    return val/10;
}
```

written by hpplayer original link here

Solution 3

Inspired by others I had to try it myself...

Solution 1, reverse and then increase while reversing back

```
def plusOne(self, head):
    tail = None
    while head:
        head.next, head, tail = tail, head.next, head
    carry = 1
    while tail:
        carry, tail.val = divmod(carry + tail.val, 10)
        if carry and not tail.next:
            tail.next = ListNode(0)
        tail.next, tail, head = head, tail.next, tail
    return head
```

Solution 2, with pure reverse helper

```
def plusOne(self, head):
    def reverse(head):
        rev = None
        while head:
            head.next, head, rev = rev, head.next, head
        return rev
    head = node = reverse(head)
    while node.val == 9:
        node.val = 0
        node.next = node = node.next or ListNode(0)
    node.val += 1
    return reverse(head)
```

If you don't like that bottom while-loop, here's a more normal way I guess:

```
    while node.val == 9:
        node.val = 0
        if not node.next:
            node.next = ListNode(0)
        node = node.next
```

written by StefanPochmann original link here

From Leetcoder.