021 Merge Two Sorted Lists

2018年3月29日　　10:57

Question:

Merge two sorted linked lists and return it as a new list. The new list should be made by splicing together the nodes of the first two lists.

合并两个已排序的链表，并将其作为一个新列表返回。新列表应该通过拼接前两个列表的节点来完成。

Example:

Input: 1->2->4, 1->3->4

Output: 1->1->2->3->4->4

来自 <https://leetcode.com/problems/merge-two-sorted-lists/description/>

Solution for Python3:

```
# Definition for singly-linked list.
# class ListNode:
#     def __init__(self, x):
#         self.val = x
#         self.next = None

Iteration Version:
class Solution:
    def mergeTwoLists(self, l1, l2):
        """
        :type l1: ListNode
        :type l2: ListNode
        :rtype: ListNode
        """
        root = ListNode(0)
        tail = root
        while l1 and l2:
            if l1.val < l2.val:
                tail.next = l1
                l1 = l1.next
            else:
                tail.next = l2
                l2 = l2.next
            tail = tail.next
        tail.next = l1 if l1 else l2
        (tail.next = l1 or l2)另一种写法
        return root.next

Recursive Version:
class Solution:
    def mergeTwoLists(self, l1, l2):
        """
        :type l1: ListNode
        :type l2: ListNode
        :rtype: ListNode
        """
        if not l1 or not l2:
            return l1 or l2
        if l1.val < l2.val:
            l1.next = self.mergeTwoLists(l1.next, l2)
            reutrn l1
        else:
            l2.next = self.mergeTwoLists(l1, l2.next)
            return l2
```

Solution for C++:

```
 /**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode(int x) : val(x), next(NULL) {}
 * };
 */
class Solution {
public:
    ListNode* mergeTwoLists(ListNode* l1, ListNode* l2) {
        ListNode root(0);
        ListNode* tail = &root;
        while (l1 && l2) {
            if (l1->val < l2->val) {
                tail->next = l1;
                l1 = l1->next;
            } else {
                tail->next = l2;
                l2 = l2->next;
            }
            tail = tail->next;
        }
        tail->next = l1 ? l1 : l2;
        return root.next;
    }
};
```

Appendix:

python的递归版本不错