

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:

```
1  # Definition for singly-linked list.
2  # class ListNode:
3  #     def __init__(self, x):
4  #         self.val = x
5  #         self.next = None
6
7  Iteration Version:
8  class Solution:
9      def mergeTwoLists(self, l1, l2):
10         """
11         :type l1: ListNode
12         :type l2: ListNode
13         :rtype: ListNode
14         """
15         root = ListNode(0)
16         tail = root
17         while l1 and l2:
18             if l1.val < l2.val:
19                 tail.next = l1
20                 l1 = l1.next
21             else:
22                 tail.next = l2
23                 l2 = l2.next
24             tail = tail.next
25         tail.next = l1 if l1 else l2
26         (tail.next = l1 or l2)另一种写法
27         return root.next
28
29 Recursive Version:
30 class Solution:
31     def mergeTwoLists(self, l1, l2):
32         """
33         :type l1: ListNode
34         :type l2: ListNode
35         :rtype: ListNode
36         """
37         if not l1 or not l2:
38             return l1 or l2
```

```

39         if l1.val < l2.val:
40             l1.next = self.mergeTwoLists(l1.next, l2)
41             return l1
42         else:
43             l2.next = self.mergeTwoLists(l1, l2.next)
44             return l2

```

Solution for C++:

```

1  /**
2   * Definition for singly-linked list.
3   * struct ListNode {
4   *     int val;
5   *     ListNode *next;
6   *     ListNode(int x) : val(x), next(NULL) {}
7   * };
8   */
9  class Solution {
10 public:
11     ListNode* mergeTwoLists(ListNode* l1, ListNode* l2) {
12         ListNode root(0);
13         ListNode* tail = &root;
14         while (l1 && l2) {
15             if (l1->val < l2->val) {
16                 tail->next = l1;
17                 l1 = l1->next;
18             } else {
19                 tail->next = l2;
20                 l2 = l2->next;
21             }
22             tail = tail->next;
23         }
24         tail->next = l1 ? l1 : l2;
25         return root.next;
26     }
27 };

```

Appendix:

python的递归版本不错