0021. 合并两个有序链表

ITCharge 大约 1 分钟

- 标签：递归、链表
- 难度：简单

题目链接

- 0021. 合并两个有序链表 - 力扣

题目大意

描述：给定两个升序链表的头节点 `list1` 和 `list2` 。

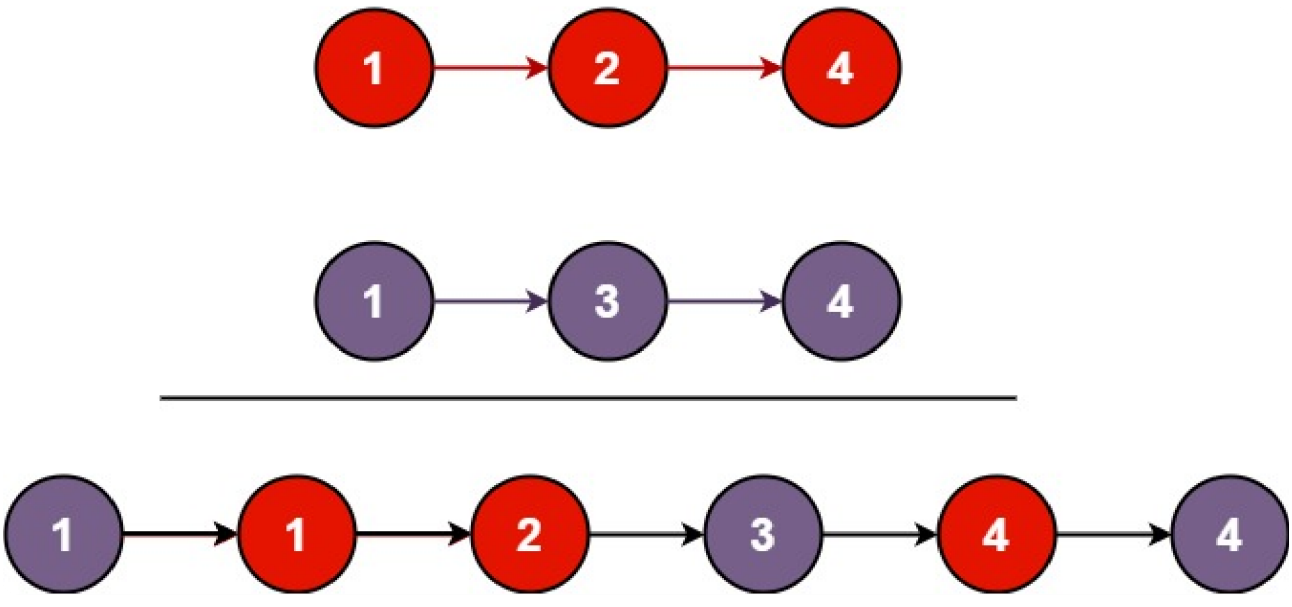
要求：将其合并为一个升序链表。

说明：

- 两个链表的节点数目范围是 $[0, 50]$
- $-100 \leq Node.val \leq 100$。
- `list1` 和 `list2` 均按 **非递减顺序** 排列

示例：

- 示例 1：



```py
输入：list1 = [1,2,4], list2 = [1,3,4]
输出：[1,1,2,3,4,4]
```

- 示例 2：

```py
输入：list1 = [], list2 = []
输出：[]
```

解题思路

思路 1：归并排序

利用归并排序的思想，具体步骤如下：

1. 使用哑节点 `dummy_head` 构造一个头节点，并使用 `curr` 指向 `dummy_head` 用于遍历。
2. 然后判断 `list1` 和 `list2` 头节点的值，将较小的头节点加入到合并后的链表中。并向后移动该链表的头节点指针。
3. 然后重复上一步操作，直到两个链 出现链表为空的情况。
4. 将剩余链表链接到合并后的链表中。
5. 将哑节点 `dummy_dead` 的下一个链节点 `dummy_head.next` 作为合并后有序链表的头节点返回。

思路 1：代码

```py
class Solution:
    def mergeTwoLists(self, list1: Optional[ListNode], list2:
Optional[ListNode]) -> Optional[ListNode]:
        dummy_head = ListNode(-1)

        curr = dummy_head
        while list1 and list2:
            if list1.val <= list2.val:
                curr.next = list1
                list1 = list1.next
            else:
                curr.next = list2
```

```
            list2 = list2.next
        curr = curr.next

    curr.next = list1 if list1 is not None else list2

    return dummy_head.next
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$。
- **空间复杂度**：$O(1)$。