0024. 两两交换链表中的节点

ITCharge 大约 1 分钟

- 标签：递归、链表
- 难度：中等

题目链接

- 0024. 两两交换链表中的节点 - 力扣

题目大意

描述：给定一个链表的头节点 `head` 。

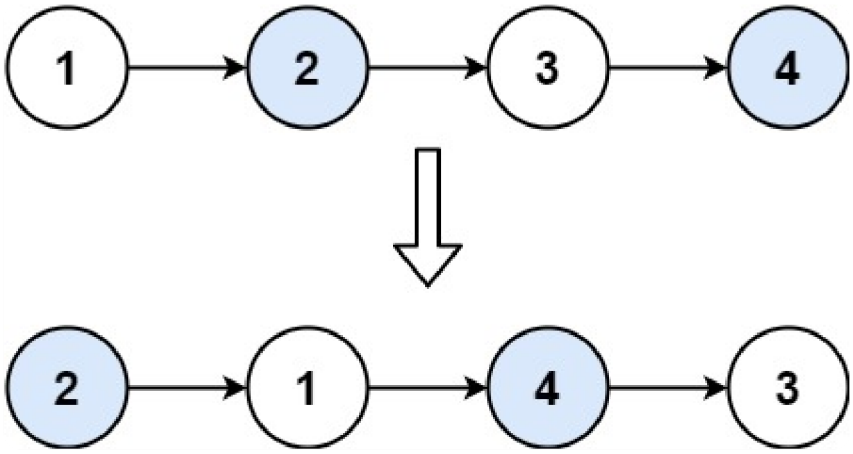
要求：按顺序将链表中每两个节点交换一下，并返回交换后的链表。

说明：

- 需要实际进行节点交换，而不是纸 :节点内部的值。
- 链表中节点的数目在范围 $[0, 100]$ 内。
- $0 \leq Node.val \leq 100$。

示例：

- 示例 1：



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

- 示例 2：

```py
输入：head = []
输出：[]
```

解题思路

思路 1：迭代

1. 创建一个哑节点 `new_head` ，令 `new_head.next = head` 。
2. 遍历链表，并判断当前链表后两位节点是否为空。如果后两个节点不为空，则使用三个指针： `curr` 指向当前节点， `node1` 指向下一个节点， `node2` 指向下面第二个节点。
3. 将 `curr` 指向 `node2` ， `node1` 指向 `node2` 后边的节点， `node2` 指向 `node1` 。则节点关系由 `curr → node1 → node2` 变为了 `curr → node2 → node1` 。
4. 依次类推，最终返回哑节点连接的下一个节点。

思路 1：代码

```py
class Solution:
    def swapPairs(self, head: ListNode) -> ListNode:
        new_head = ListNode(0)
        new_head.next = head
        curr = new_head
        while curr.next and curr.next.next:
            node1 = curr.next
            node2 = curr.next.next
            curr.next = node2
            node1.next = node2.next
            node2.next = node1
            curr = node1
        return new_head.next
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$，其中 n 为链表的节点数量。
- **空间复杂度**：$O(n)$。