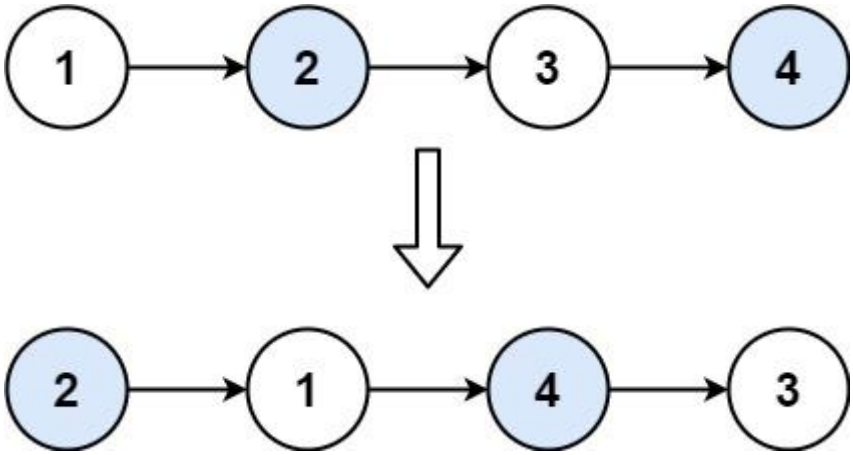
24. Swap Nodes in Pairs

Given a linked list, swap every two adjacent nodes and return its head. You must solve the problem without modifying the values in the list's nodes (i.e., only nodes themselves may be changed.)

Example 1:




```
Input: head = [1,2,3,4]
Output: [2,1,4,3]
```

Example 2:

```
Input: head = []
Output: []
```

Example 3:

```
Input: head = [1]
Output: [1]
```

Constraints:

- The number of nodes in the list is in the range `[0, 100]`.
- `0 <= Node.val <= 100`

Code:

```
# Definition for singly-linked list.
# class ListNode:
#     def __init__(self, val=0, next=None):
#         self.val = val
#         self.next = next
class Solution:
    def swapPairs(self, head: Optional[ListNode]) -> Optional[ListNode]:

        if not head or not head.next:
            return head

        first_node = head
        second_node = head.next

        first_node.next = self.swapPairs(second_node.next)
        second_node.next = first_node

        return second_node
```