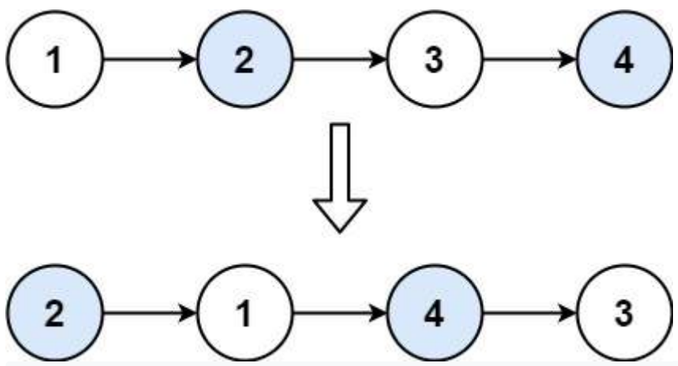
Description | Solution | Discuss (999+) | Submissions

24. Swap Nodes in Pairs

Medium 5726 273 Add to List Share

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`

Java | Autocomplete

```java
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next; }
 * }
 */
class Solution {
    public ListNode swapPairs(ListNode head) {

        if(head == null)
            return null;

        ListNode firstNode = head;
        ListNode secondNode = head.next;

        while(secondNode != null){
            int temp = firstNode.val;
            firstNode.val = secondNode.val;
            secondNode.val = temp;

            if(secondNode.next != null){
                firstNode = secondNode.next;
                secondNode = secondNode.next.next;
            }else{
                secondNode = null;
            }
        }
```

Testcase | Run Code Result | Debugger

Accepted Runtime: 0 ms

Your input `[1,2,3,4]`

Output `[2,1,4,3]` Diff

Expected `[2,1,4,3]`

Problems | Pick One | Prev | 127 | Next | Console | Use Example Testcases | Run Code | Submit