연결 리스트를 홀수번째 노드 다음에 짝수번째 노드가 오도록 재구성하라. 공간 복잡도 O(1), 시간 복잡도 O(n)에 풀이하라.

Input: 1->2->3->4->5->NULL
Output: 1->3->5->2->4->NULL

```
# Definition for singly-linked list.
# class ListNode:
#     def __init__(self, val=0, next=None):
#         self.val = val
#         self.next = next
```

1.반복

```
class Solution:
    def oddEvenList(self, head: ListNode) -> ListNode:
        if not head:
            return head
        odd, even, even_head = head, head.next, head.next
        while even and even.next:
            odd.next, even.next = odd.next.next, even.next.next
            odd, even = odd.next, even.next
        odd.next = even_head
        return head
```

```
class Solution:
    def oddEvenList(self, head: ListNode) -> ListNode:
        result = odd = head
        prev = None
        count = 1
        while head:
            if count % 2 and prev:
                odd.next, prev.next, head.next, head, prev, odd = head, head.next, odd.next,
head.next, head, head
                count += 1
                continue
            prev, head = head, head.next
            count += 1
        return result
```