Problem 25: Reverse Nodes in k-Group

Problem Information

Difficulty: Hard
Acceptance Rate: 64.59%
Paid Only: No
Tags: Linked List, Recursion

Problem Description

Given the `head` of a linked list, reverse the nodes of the list `k` at a time, and return _the modified list_.

`k` is a positive integer and is less than or equal to the length of the linked list. If the number of nodes is not a multiple of `k` then left-out nodes, in the end, should remain as it is.

You may not alter the values in the list's nodes, only nodes themselves may be changed.

Example 1:

Input: head = [1,2,3,4,5], k = 2 **Output:** [2,1,4,3,5]

Example 2:

Input: head = [1,2,3,4,5], k = 3 **Output:** [3,2,1,4,5]

Constraints:

* The number of nodes in the list is `n`. * `1 <= k <= n <= 5000` * `0 <= Node.val <= 1000`

Follow-up: Can you solve the problem in `O(1)` extra memory space?

Code Snippets

C++:

```
/**
* Definition for singly-linked list.
* struct ListNode {
* int val;
* ListNode *next;
* ListNode() : val(0), next(nullptr) {}
* ListNode(int x) : val(x), next(nullptr) {}
* ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
ListNode* reverseKGroup(ListNode* head, int k) {

}
};
```

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 reverseKGroup(ListNode head, int k) {

}
}
```

Python3:

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