class ListNode:

def \_\_init\_\_(self, val=0, next=None):

self.val = val

self.next = next

class Solution:

def reverseList(self, head, k):

prev = None

curr = head

for \_ in range(k):

next\_node = curr.next

curr.next = prev

prev = curr

curr = next\_node

return prev, curr # Return new head and the next node after k elements

def reverseKGroup(self, head, k):

dummy = ListNode(0)

dummy.next = head

ptr = dummy

while ptr is not None:

tracker = ptr

# Check if there are k nodes to reverse

for \_ in range(k):

tracker = tracker.next

if tracker is None:

return dummy.next # If fewer than k nodes, return result

# Reverse k nodes

prev, curr = self.reverseList(ptr.next, k)

last\_node\_of\_reversed\_group = ptr.next

last\_node\_of\_reversed\_group.next = curr

ptr.next = prev

ptr = last\_node\_of\_reversed\_group # Move ptr forward

return dummy.next