Sumeet Shailendra Kadam

23

In [2]:

```python
class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

def rotateRight(head: ListNode, k: int) -> ListNode:
    if not head or not head.next or k == 0:
        return head

    # Compute the length of the list
    length = 1
    tail = head
    while tail.next:
        tail = tail.next
        length += 1

    # Make the list circular
    tail.next = head

    # Find the new tail position
    k = k % length
    steps_to_new_tail = length - k
    new_tail = head
    for _ in range(steps_to_new_tail - 1):
        new_tail = new_tail.next

    # Break the circular link
    new_head = new_tail.next
    new_tail.next = None

    return new_head

# Helper function to convert list to linked list
def list_to_linked_list(arr):
    if not arr:
        return None
    head = ListNode(arr[0])
    current = head
    for val in arr[1:]:
        current.next = ListNode(val)
        current = current.next
    return head

# Helper function to convert linked list to list
def linked_list_to_list(head):
    result = []
    while head:
        result.append(head.val)
        head = head.next
    return result

# Example usage:
head = list_to_linked_list([1, 2, 3, 4, 5])
k = 2
print("Original List:", linked_list_to_list(head))
rotated_head = rotateRight(head, k)
print("Rotated List:", linked_list_to_list(rotated_head))  # Output: [4, 5, 1, 2, 3]
```

```
Original List: [1, 2, 3, 4, 5]
Rotated List: [4, 5, 1, 2, 3]
```

In [ ]: