ASSIGNMENT 03

Varun Dhanak 12

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

# Updated Solution with an example
class Solution(object):
    def rotateRight(self, head, k):
        if not head or not head.next or k == 0:
            return head

        # Calculate the length of the list and identify the tail node
        length, tail = 1, head
        while tail.next:
            tail = tail.next
            length += 1  # Fixed the syntax issue

        # Handle cases where k is greater than or equal to the length
        k %= length
        if k == 0:
            return head

        # Find the new head after rotation
        cur = head
        for i in range(length - k - 1):
            cur = cur.next

        # Reassign pointers to rotate the list
        newHead = cur.next
        cur.next = None
        tail.next = head

        return newHead

# Example function to test the solution
def print_linked_list(head):
    """Helper function to print the linked list."""
    result = []
    while head:
        result.append(head.val)
        head = head.next
```

```
    print(result)

# Create an example linked list: [1, 2, 3, 4, 5]
node1 = ListNode(1)
node2 = ListNode(2)
node3 = ListNode(3)
node4 = ListNode(4)
node5 = ListNode(5)

node1.next = node2
node2.next = node3
node3.next = node4
node4.next = node5

# Initialize the solution and rotate the linked list
solution = Solution()
k = 2  # Number of rotations
rotated_head = solution.rotateRight(node1, k)

# Print the result
print("Original list: [1, 2, 3, 4, 5]")
print("After rotating by {}:".format(k), end=" ")
print_linked_list(rotated_head)
```

```
Original list: [1, 2, 3, 4, 5]
After rotating by 2: [4, 5, 1, 2, 3]
```