Java Rotate Linked List

write java code for the Given the head of a linked list, rotate the list to the right by k places.
Input: head = [1,2,3,4,5], k = 2
Output: [4,5,1,2,3]
use method public ListNode rotateRight(ListNode head, int k) {

  }

Here's a Java implementation for the `rotateRight` method:

```java
public class ListNode {
    int val;
    ListNode next;
    ListNode(int val) { this.val = val; }
}

public class Solution {
    public ListNode rotateRight(ListNode head, int k) {
        if (head == null || head.next == null || k == 0) {
            return head;
        }

        // Step 1: Find the length of the linked list
        ListNode current = head;
        int length = 1;
        while (current.next != null) {
            current = current.next;
            length++;
        }

        // Step 2: Connect the last node to the head to make it a circular list
        current.next = head;

        // Step 3: Calculate the number of moves needed after k rotations
        k = k % length;
        int moveSteps = length - k;

        // Step 4: Find the new tail and new head
        ListNode newTail = head;
        for (int i = 1; i < moveSteps; i++) {
            newTail = newTail.next;
        }

        ListNode newHead = newTail.next;

        // Step 5: Break the circular list
        newTail.next = null;

        return newHead;
    }

    // Helper method to print the list (for testing purposes)
    public static void printList(ListNode head) {
        ListNode current = head;
        while (current != null) {
            System.out.print(current.val + " ");
            current = current.next;
        }
        System.out.println();
    }

    public static void main(String[] args) {
        // Creating the list: [1, 2, 3, 4, 5]
        ListNode head = new ListNode(1);
        head.next = new ListNode(2);
        head.next.next = new ListNode(3);
        head.next.next.next = new ListNode(4);
        head.next.next.next.next = new ListNode(5);

        Solution solution = new Solution();
        ListNode rotatedHead = solution.rotateRight(head, 2);

        // Print the result: [4, 5, 1, 2, 3]
        printList(rotatedHead);
    }
}
```

Explanation:

1. **Step 1**: Calculate the length of the linked list.
2. **Step 2**: Create a circular list by connecting the last node to the head.
3. **Step 3**: Calculate the effective number of rotations by using `k % length` (if `k` is greater than the length).
4. **Step 4**: Find the new tail (which will be the node before the new head after rotation).
5. **Step 5**: Break the circular connection and return the new head.

ChatGPT can make mistakes. Check important info.