Split Linked List in K parts LeetCode

You are given a singly linked list and an integer k. Your task is to split the linked list into k parts such that each part contains approximately equal number of nodes. If there are remaining nodes, they should be distributed as evenly as possible among the parts. Implement a program to achieve this split and print the resulting linked list parts.

Example: Input: Linked List: 1 -> 2 -> 3 -> 4 -> 5 -> 6, k = 4

Output:

- Linked List 1: 1 -> 2
- Linked List 2: 3 -> 4
- Linked List 3: 5
- Linked List 4: 6

Approach 1: Split the linked list into k parts

1. Calculate the length of the linked list.
2. Determine the size of each group (**groupSize**) as **length / k** and the number of remaining groups (**remGroup**) as **length % k**.
3. Initialize a result vector **res** of size **k** to store the head nodes of each group.
4. Iterate through the linked list and dynamically create groups:
    - Create a group with **groupSize** nodes.
    - Distribute remaining nodes if **remGroup** is not zero.
    - Set the tail of the current group to null.
    - Store the current group in the result vector.
5. Return the result vector containing the head nodes of the split groups.
6. **Time Complexity:**
    - Calculating the length of the linked list: O(n)
    - Iterating through the linked list to create groups: O(n)
    - **The overall time complexity is O(n).**
7. **Space Complexity:**
    - Additional space used for pointers and variables: O(1)
    - The result vector res of size k: O(k)
    - **The overall space complexity is O(k).**