AlgoExpert Quad Layout C# 12px Sublime Monok

Prompt Scratchpad Our Solution(s) Video Explanation Run Code

Solution 1

```
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

public class Program {
  // O(n) time | O(1) space - where n is the number of nodes in the Linked List
  public static LinkedList RearrangeLinkedList(LinkedList head, int k) {
    LinkedList smallerListHead = null;
    LinkedList smallerListTail = null;
    LinkedList equalListHead = null;
    LinkedList equalListTail = null;
    LinkedList greaterListHead = null;
    LinkedList greaterListTail = null;

    LinkedList node = head;
    while (node != null) {
      if (node.value < k) {
        LinkedListPair smallerList =
          growLinkedList(smallerListHead, smallerListTail, node);
        smallerListHead = smallerList.head;
        smallerListTail = smallerList.tail;
      } else if (node.value > k) {
        LinkedListPair greaterList =
          growLinkedList(greaterListHead, greaterListTail, node);
        greaterListHead = greaterList.head;
        greaterListTail = greaterList.tail;
      } else {
        LinkedListPair equalList =
          growLinkedList(equalListHead, equalListTail, node);
        equalListHead = equalList.head;
        equalListTail = equalList.tail;
      }

      LinkedList prevNode = node;
      node = node.next;
      prevNode.next = null;
    }

    LinkedListPair firstPair = connectLinkedLists(smallerListHead, smallerListTail,
        equalListHead, equalListTail);
    LinkedListPair finalPair = connectLinkedLists(firstPair.head, firstPair.tail,
        greaterListHead, greaterListTail);
    return finalPair.head;
  }

  public static LinkedListPair growLinkedList(LinkedList head, LinkedList tail,
    LinkedList node) {
    LinkedList newHead = head;
    LinkedList newTail = node;

    if (newHead == null) newHead = node;
    if (tail != null) tail.next = node;

    return new LinkedListPair(newHead, newTail);
  }

  public static LinkedListPair connectLinkedLists(LinkedList headOne, LinkedList tailOne,
    LinkedList headTwo, LinkedList tailTwo) {
    LinkedList newHead = headOne == null ? headTwo : headOne;
    LinkedList newTail = tailTwo == null ? tailOne : tailTwo;

    if (tailOne != null) tailOne.next = headTwo;

    return new LinkedListPair(newHead, newTail);
  }

  public class LinkedListPair {
    public LinkedList head;
    public LinkedList tail;

    public LinkedListPair(LinkedList head, LinkedList tail) {
      this.head = head;
      this.tail = tail;
    }
  }

  public class LinkedList {
    public int value;
    public LinkedList next;

    public LinkedList(int value) {
      this.value = value;
      next = null;
    }
  }
}
```