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

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

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

  // O(n + m) time | O(n + m) space - where n is the number of nodes in the first
  // Linked List and m is the number of nodes in the second Linked List
  public static LinkedList mergeLinkedLists(LinkedList headOne, LinkedList headTwo) {
    recursiveMerge(headOne, headTwo, null);
    return headOne.value < headTwo.value ? headOne : headTwo;
  }

  public static void recursiveMerge(LinkedList p1, LinkedList p2, LinkedList p1Prev) {
    if (p1 == null) {
      p1Prev.next = p2;
      return;
    }
    if (p2 == null) return;

    if (p1.value < p2.value) {
      recursiveMerge(p1.next, p2, p1);
    } else {
      if (p1Prev != null) p1Prev.next = p2;
      LinkedList newP2 = p2.next;
      p2.next = p1;
      recursiveMerge(p1, newP2, p2);
    }
  }
}

```