Our Solution(s) Run Code

Solution 1 Solution 2

```python
# Copyright © 2020 AlgoExpert, LLC. All rights reserved.

class LinkedList:
    def __init__(self, value):
        self.value = value
        self.next = None


# O(n + m) time | O(n + m) space - where n is the number of nodes in
# Linked List and m is the number of nodes in the second Linked List
def mergeLinkedLists(headOne, headTwo):
    recursiveMerge(headOne, headTwo, None)
    return headOne if headOne.value < headTwo.value else headTwo


def recursiveMerge(p1, p2, p1Prev):
    if p1 is None:
        p1Prev.next = p2
        return
    if p2 is None:
        return

    if p1.value < p2.value:
        recursiveMerge(p1.next, p2, p1)
    else:
        if p1Prev is not None:
            p1Prev.next = p2
        newP2 = p2.next
        p2.next = p1
        recursiveMerge(p1, newP2, p2)

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```python
# This is an input class. Do not edit.
class LinkedList:
    def __init__(self, value):
        self.value = value
        self.next = None


def mergeLinkedLists(headOne, headTwo):
    # Write your code here.
    pass

```

Our Tests



Custom Output Submit Code

```
        while current.next is not None:
            current = current.next
        for value in values:
            current.next = ListNode(value)
            current = current.next
        return self

    def getNodeValue(self):
        nodes = []
        current = self
        while current is not None:
            nodes.append(current.value)
            current = current.next
        return nodes
```

**Run or submit code when you're ready.**