Our Solution(s) Run Code

Solution 1 Solution 2

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

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

// O(n + m) time | O(1) space - where n is the number of nodes in the
// Linked List and m is the number of nodes in the second Linked List
function mergeLinkedLists(headOne, headTwo) {
  let p1 = headOne;
  let p1Prev = null;
  let p2 = headTwo;
  while (p1 !== null && p2 !== null) {
    if (p1.value < p2.value) {
      p1Prev = p1;
      p1 = p1.next;
    } else {
      if (p1Prev !== null) p1Prev.next = p2;
      p1Prev = p2;
      p2 = p2.next;
      p1Prev.next = p1;
    }
  }
  if (p1 === null) p1Prev.next = p2;
  return headOne.value < headTwo.value ? headOne : headTwo;
}

exports.LinkedList = LinkedList;
exports.mergeLinkedLists = mergeLinkedLists;

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```
// This is an input class. Do not edit.
class LinkedList {
  constructor(value) {
    this.value = value;
    this.next = null;
  }
}

function mergeLinkedLists(headOne, headTwo) {
  // Write your code here.
}

// Do not edit the line below.
exports.LinkedList = LinkedList;
exports.mergeLinkedLists = mergeLinkedLists;

```

Our Tests

Custom Output Submit Code

```
}

addMany(values) {
  let current = this;
  while (current.next !== null) {
    current = current.next;
  }
  for (const value of values) {
    current.next = new LinkedList(value);
    current = current.next;
  }
  return this;
}

getNodesInArray() {
  const nodes = [];
```

Run or submit code when you're ready.