Our Solution(s) Run Code

Solution 1 Solution 2

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

class BinaryTree {
  constructor(value) {
    this.value = value;
    this.left = null;
    this.right = null;
  }
}

// O(n) time | O(d) space - where n is the number of nodes in the Bin
function flattenBinaryTree(root) {
  const [leftMost, _] = flattenTree(root);
  return leftMost;
}

function flattenTree(node) {
  let leftMost, rightMost;

  if (node.left === null) {
    leftMost = node;
  } else {
    const [leftSubtreeLeftMost, leftSubtreeRightMost] = flattenTree(no
    connectNodes(leftSubtreeRightMost, node);
    leftMost = leftSubtreeLeftMost;
  }

  if (node.right === null) {
    rightMost = node;
  } else {
    const [rightSubtreeLeftMost, rightSubtreeRightMost] = flattenTree(
    connectNodes(node, rightSubtreeLeftMost);
    rightMost = rightSubtreeRightMost;
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```
// This is the class of the input root. Do not edit it.
class BinaryTree {
  constructor(value) {
    this.value = value;
    this.left = null;
    this.right = null;
  }
}

function flattenBinaryTree(root) {
  // Write your code here.
}

// Do not edit the lines below.
exports.BinaryTree = BinaryTree;
exports.flattenBinaryTree = flattenBinaryTree;

```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.