Our Solution(s) Run Code

Solution 1 Solution 2

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

class Program {
    class BinaryTree {
        var value: Int
        var left: BinaryTree?
        var right: BinaryTree?

        init(value: Int) {
            self.value = value
            left = nil
            right = nil
        }
    }

    // O(n) time | O(n) space - where n is the number of nodes
    // in the Binary Tree
    func flattenBinaryTree(root: BinaryTree) -> BinaryTree {
        var inOrderNodes = [BinaryTree]()
        getNodesInOrder(root: root, array: &inOrderNodes)
        for i in 0 ..< inOrderNodes.count - 1 {
            var leftNode = inOrderNodes[i]
            var rightNode = inOrderNodes[i + 1]
            leftNode.right = rightNode
            rightNode.left = leftNode
        }
        return inOrderNodes[0]
    }

    func getNodesInOrder(root: BinaryTree?, array: inout [BinaryTree])
        if let tree = root {
            getNodesInOrder(root: tree.left, array: &array)
            array.append(tree)
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```swift
class Program {
    // This is the class of the input root. Do not edit it.
    class BinaryTree {
        var value: Int
        var left: BinaryTree?
        var right: BinaryTree?

        init(value: Int) {
            self.value = value
            left = nil
            right = nil
        }
    }

    func flattenBinaryTree(root: BinaryTree) -> BinaryTree {
        // Write your code here.
        return root
    }
}

```

Our Tests

Custom Output Submit Code

```
        let expected = [3, 3]
        Prog.assertEqual(expected, actual)
    }
    runTest("Test Case 4") { () throws -> Void in
        let root = TestBinaryTree(value: 5).insert(value: 10)
        let leftMostNode = program.findLeastBinary(tree: root)
        let actual = leftRightTraversal(tree: leftMostNode)
        let expected = [5, 5, 5, 5]
        Prog.assertEqual(expected, actual)
    }
    runTest("Test Case 5") { () throws -> Void in
        let root = TestBinaryTree(value: 5).insert(value: 10).
        let leftMostNode = program.findLeastBinary(tree: root)
        let actual = leftRightTraversal(tree: leftMostNode)
        let expected = [5, 5, 5, 5, 5, 5]
```

Run or submit code when you're ready.