

Our Solution(s)

Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     class BinaryTree {
5         var value: Int
6         var left: BinaryTree?
7         var right: BinaryTree?
8
9         init(value: Int) {
10             self.value = value
11             left = nil
12             right = nil
13         }
14     }
15
16     // O(n) time | O(n) space - where n is the number of nodes
17     // in the Binary Tree
18     func flattenBinaryTree(root: BinaryTree) -> BinaryTree {
19         var inOrderNodes = [BinaryTree]()
20         getNodesInOrder(root: root, array: &inOrderNodes)
21         for i in 0 ..< inOrderNodes.count - 1 {
22             var leftNode = inOrderNodes[i]
23             var rightNode = inOrderNodes[i + 1]
24             leftNode.right = rightNode
25             rightNode.left = leftNode
26         }
27         return inOrderNodes[0]
28     }
29
30     func getNodesInOrder(root: BinaryTree?, array: inout [BinaryTree])
31     if let tree = root {
32         getNodesInOrder(root: tree.left, array: &array)
33         array.append(tree)
```

Your Solutions

Run Code

Solution 1Solution 2Solution 3

```
1 class Program {
2     // This is the class of the input root. Do not edit it.
3     class BinaryTree {
4         var value: Int
5         var left: BinaryTree?
6         var right: BinaryTree?
7
8         init(value: Int) {
9             self.value = value
10            left = nil
11            right = nil
12        }
13    }
14
15    func flattenBinaryTree(root: BinaryTree) -> BinaryTree {
16        // Write your code here.
17        return root
18    }
19 }
20
```

Our Tests

```
1 class ProgramTests {
2     func test() {
3         let program = Program()
4         let root = BinaryTree(value: 1)
5         root.left = BinaryTree(value: 2)
6         root.right = BinaryTree(value: 3)
7         root.left!.right = BinaryTree(value: 4)
8         root.left!.left = BinaryTree(value: 5)
9         root.left!.left!.left = BinaryTree(value: 6)
10        root.left!.left!.left!.left = BinaryTree(value: 7)
11    }
12 }
```

Custom Output

Submit Code

```

14     expected = [0, 0]
15     return expected(expecteds, actual)
16
17
18 def test_test_case_2():
19     test = TestCasesTestCase(2, expecteds=[0])
20     actual = program.TestCasesTestCase(test)
21     expected = [0, 0, 0, 0, 0]
22     return expected(expecteds, actual)
23
24
25 def test_test_case_3():
26     test = TestCasesTestCase(3, expecteds=[0])
27     actual = program.TestCasesTestCase(test)
28     expected = [0, 0, 0, 0, 0, 0]
29
30
31

```

Run or submit code when you're ready.