

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
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(d) space - where n is the number of nodes in
17     // the Binary Tree and d is the depth (height) of the Binary Tree
18     func rightSiblingTree(root: BinaryTree) -> BinaryTree {
19         mutate(node: root, parent: nil, isLeftChild: false)
20         return root
21     }
22
23     func mutate(node: BinaryTree?, parent: BinaryTree?, isLeftChild: Boolean) {
24         if let tree = node {
25             var left = tree.left
26             var right = tree.right
27             mutate(node: left, parent: tree, isLeftChild: true)
28             if let p = parent {
29                 if isLeftChild {
30                     tree.right = p.right
31                 } else {
32                     if let right = p.right {
33                         tree.right = right.left
34                     }
35                 }
36             }
37         }
38     }
39 }
```

Solution 1   Solution 2   Solution 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 rightSiblingTree(root: BinaryTree) -> BinaryTree {
16        // Write your code here.
17        return root
18    }
19 }
20
```

Our Tests

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

Custom Output

Submit Code

```
1
2
3
4
5
6
7
8
9
10
11
12
```

```

14     expected = 10
15     #q = await asyncio.sleep(100, expected)
16
17
18     #assert "Test case 2" in str(expected) == "Test 20"
19     #q = asyncio.sleep(100, 20, expected=100)
20     #assert "Test case 2" in str(expected) == "Test 20"
21     #assert == get_test_case(2, expected)
22     #assert == 20, 10
23     #q = await asyncio.sleep(100, expected)
24
25
26     #assert "Test case 2" in str(expected) == "Test 20"
27     #q = asyncio.sleep(100, 20, expected=100)
28     #assert "Test case 2" in str(expected) == "Test 20"
29     #assert == get_test_case(2, expected)
30     #assert == 20, 10, 10

```

Run or submit code when you're ready.