

PromptScratchpadOur Solution(s)Video Explanation

Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     class BST {
5         var value: Int
6         var left: BST?
7         var right: BST?
8
9         init(value: Int) {
10             self.value = value
11             left = nil
12             right = nil
13         }
14
15         // Average: O(log(n)) time | O(1) space
16         // Worst: O(n) time | O(1) space
17         func insert(value: Int) -> BST {
18             var currentNode: BST? = self
19
20             while true {
21                 if let node = currentNode, value < node.value {
22                     if node.left === nil {
23                         node.left = BST(value: value)
24                         break
25                     } else {
26                         currentNode = node.left
27                     }
28                 } else if let node = currentNode {
29                     if node.right === nil {
30                         node.right = BST(value: value)
31                         break
32                     } else {
33                         currentNode = node.right
34                     }
35                 }
36             }
37
38             return self
39         }
40
41         // Average: O(log(n)) time | O(1) space
42         // Worst: O(n) time | O(1) space
43         func contains(value: Int) -> Bool {
44             var currentNode: BST? = self
45
46             while currentNode !== nil {
47                 if let node = currentNode, value < node.value {
48                     currentNode = node.left
49                 } else if let node = currentNode, value > node.value {
50                     currentNode = node.right
51                 } else {
52                     return true
53                 }
54             }
55
56             return false
57         }
58
59         // Average: O(log(n)) time | O(1) space
60         // Worst: O(n) time | O(1) space
61         func remove(value: Int, parentNode: BST?) -> BST {
62             var currentNode: BST? = self
63             var parentNode: BST? = parentNode
64             while let node = currentNode {
65                 if value < node.value {
66                     parentNode = node
67                     currentNode = node.left
68                 } else if value > node.value {
69                     parentNode = node
70                     currentNode = node.right
71                 } else {
72                     if let left = node.left, let right = node.right {
73                         node.value = right.getMinValue()
74                         right.remove(value: node.value, parentNode: node)
75                     } else if parentNode === nil {
76                         if let left = node.left {
77                             node.value = left.value
78                             node.right = left.right
79                             node.left = left.left
80                         } else if let right = node.right {
81                             node.value = right.value
82                             node.left = right.left
83                             node.right = right.right
84                         } else {
85                             // This is a single-node tree; do nothing.
```