

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 package main
4
5 type BST struct {
6     Value int
7
8     Left *BST
9     Right *BST
10 }
11
12 // Average: O(log(n)) time | O(1) space
13 // Worst: O(n) time | O(1) space
14 func (tree *BST) Insert(value int) *BST {
15     current := tree
16     for {
17         if value < current.Value {
18             if current.Left == nil {
19                 current.Left = &BST{Value: value}
20                 break
21             } else {
22                 current = current.Left
23             }
24         } else {
25             if current.Right == nil {
26                 current.Right = &BST{Value: value}
27                 break
28             } else {
29                 current = current.Right
30             }
31         }
32     }
33     return tree
34 }
35
36 // Average: O(log(n)) time | O(1) space
37 // Worst: O(n) time | O(1) space
38 func (tree *BST) Contains(value int) bool {
39     current := tree
40     for current != nil {
41         if value < current.Value {
42             current = current.Left
43         } else if value > current.Value {
44             current = current.Right
45         } else {
46             return true
47         }
48     }
49     return false
50 }
51
52 // Average: O(log(n)) time | O(1) space
53 // Worst: O(n) time | O(1) space
54 func (tree *BST) Remove(value int) *BST {
55     tree.remove(value, nil)
56     return tree
57 }
58
59 func (tree *BST) remove(value int, parent *BST) {
60     current := tree
61     for current != nil {
62         if value < current.Value {
63             parent = current
64             current = current.Left
65         } else if value > current.Value {
66             parent = current
67             current = current.Right
68         } else {
69             if current.Left != nil && current.Right != nil {
70                 current.Value = current.Right.getMinValue()
71                 current.Right.remove(current.Value, current)
72             } else if parent == nil {
73                 if current.Left != nil {
74                     current.Value = current.Left.Value
75                     current.Right = current.Left.Right
76                     current.Left = current.Left.Left
77                 } else if current.Right != nil {
78                     current.Value = current.Right.Value
79                     current.Left = current.Right.Left
80                     current.Right = current.Right.Right
81                 } else {
82                     // This is a single-node tree; do nothing.
83                 }
84             } else if parent.Left == current {
85                 if current.Left != nil {
86                     parent.Left = current.Left
87                 } else {
88                     parent.Left = current.Right
89                 }
90             } else if parent.Right == current {
```

Solution 1Solution 2Solution 3

```
1 package main
2
3 // Do not edit the class below except for
4 // the insert, contains, and remove methods.
5 // Feel free to add new properties and methods
6 // to the class.
7 type BST struct {
8     Value int
9
10    Left *BST
11    Right *BST
12 }
13
14 func (tree *BST) Insert(value int) *BST {
15     // Write your code here.
16     // Do not edit the return statement of this method.
17     return tree
18 }
19
20 func (tree *BST) Contains(value int) bool {
21     // Write your code here.
22     return false
23 }
24
25 func (tree *BST) Remove(value int) *BST {
26     // Write your code here.
27     // Do not edit the return statement of this method.
28     return tree
29 }
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Custom OutputRaw OutputSubmit Code

```
91         if current.Left != nil {
92             parent.Right = current.Left
93         } else {
94             parent.Right = current.Right
95         }
96     }
97     break
98 }
99 }
100 }
101
102 func (tree *BST) getMinValue() int {
103     if tree.Left == nil {
104         return tree.Value
105     }
106     return tree.Left.getMinValue()
107 }
108 }
```

Run or submit code when you're ready.