

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     static class BST {
5         public int value;
6         public BST left;
7         public BST right;
8
9         public BST(int value) {
10             this.value = value;
11         }
12
13         // Average: O(log(n)) time | O(log(n)) space
14         // Worst: O(n) time | O(n) space
15         public BST insert(int value) {
16             if (value < this.value) {
17                 if (left == null) {
18                     BST newBST = new BST(value);
19                     left = newBST;
20                 } else {
21                     left.insert(value);
22                 }
23             } else {
24                 if (right == null) {
25                     BST newBST = new BST(value);
26                     right = newBST;
27                 } else {
28                     right.insert(value);
29                 }
30             }
31             return this;
32         }
33
34         // Average: O(log(n)) time | O(log(n)) space
35         // Worst: O(n) time | O(n) space
36         public boolean contains(int value) {
37             if (value < this.value) {
38                 if (left == null) {
39                     return false;
40                 } else {
41                     return left.contains(value);
42                 }
43             } else if (value > this.value) {
44                 if (right == null) {
45                     return false;
46                 } else {
47                     return right.contains(value);
48                 }
49             } else {
50                 return true;
51             }
52         }
53
54         // Average: O(log(n)) time | O(log(n)) space
55         // Worst: O(n) time | O(n) space
56         public BST remove(int value) {
57             remove(value, null);
58             return this;
59         }
60
61         public void remove(int value, BST parent) {
62             if (value < this.value) {
63                 if (left != null) {
64                     left.remove(value, this);
65                 }
66             } else if (value > this.value) {
67                 if (right != null) {
68                     right.remove(value, this);
69                 }
70             } else {
71                 if (left != null && right != null) {
72                     this.value = right.getMinValue();
73                     right.remove(this.value, this);
74                 } else if (parent == null) {
75                     if (left != null) {
76                         this.value = left.value;
77                         right = left.right;
78                         left = left.left;
79                     } else if (right != null) {
80                         this.value = right.value;
81                         left = right.left;
82                         right = right.right;
83                     } else {
84                         // This is a single-node tree; do nothing.
85                     }
86                 } else if (parent.left == this) {
87                     parent.left = left != null ? left : right;
88                 } else if (parent.right == this) {
89                     parent.right = left != null ? left : right;
90                 }
91             }
92         }
93     }
94 }
```

Solution 1Solution 2Solution 3

```
1 class Program {
2     static class BST {
3         public int value;
4         public BST left;
5         public BST right;
6
7         public BST(int value) {
8             this.value = value;
9         }
10
11         public BST insert(int value) {
12             // Write your code here.
13             // Do not edit the return statement of this method.
14             return this;
15         }
16
17         public boolean contains(int value) {
18             // Write your code here.
19             return false;
20         }
21
22         public BST remove(int value) {
23             // Write your code here.
24             // Do not edit the return statement of this method.
25             return this;
26         }
27     }
28 }
29
```

Custom OutputRaw OutputSubmit Code

```
91     }
92 }
93
94 public int getMinValue() {
95     if (left == null) {
96         return this.value;
97     } else {
98         return left.getMinValue();
99     }
100 }
101 }
102 }
103
```

Run or submit code when you're ready.