

Prompt	Scratchpad	Our Solution(s)	Video Explanation	Run Code	Your Solutions	Run Code
--------	------------	-----------------	-------------------	----------	----------------	----------

Solution 1Solution 2

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

Solution 1Solution 2Solution 3

```
1 public class Program {
2     public 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 bool 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         currentNode.right;
92     } else if (parentNode.right == currentNode) {
93         parentNode.right = currentNode.left !=
94             null ? currentNode.left :
95             currentNode.right;
96     }
97     break;
98 }
99 }
100 }
101
102 public int getMinValue() {
103     if (left == null) {
104         return value;
105     } else {
106         return left.getMinValue();
107     }
108 }
109 }
110 }
111 }
```

Run or submit code when you're ready.