

Solution 1

Solution 2

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

Solution 1

Solution 2

Solution 3

```
1 // Do not edit the class below except for
2 // the insert, contains, and remove methods.
3 // Feel free to add new properties and methods
4 // to the class.
5 class BST {
6   constructor(value) {
7     this.value = value;
8     this.left = null;
9     this.right = null;
10  }
11
12  insert(value) {
13    // Write your code here.
14    // Do not edit the return statement of this method.
15    return this;
16  }
17
18  contains(value) {
19    // Write your code here.
20  }
21
22  remove(value) {
23    // Write your code here.
24    // Do not edit the return statement of this method.
25    return this;
26  }
27 }
28
29 // Do not edit the line below.
30 exports.BST = BST;
31
```

Custom Output

Raw Output

Submit Code

```
91 }  
92 }  
93  
94 exports.BST = BST;  
95
```

Run or submit code when you're ready.