

Our Solution(s)

Run Code

Solution 1

Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4 using namespace std;
5
6 class BST {
7 public:
8     int value;
9     BST *left;
10    BST *right;
11
12    BST(int val) {
13        value = val;
14        left = NULL;
15        right = NULL;
16    }
17
18    // Average: O(log(n)) time | O(log(n)) space
19    // Worst: O(n) time | O(n) space
20    BST &insert(int val) {
21        if (val < value) {
22            if (left == NULL) {
23                BST *newBST = new BST(val);
24                left = newBST;
25            } else {
26                left->insert(val);
27            }
28        } else {
29            if (right == NULL) {
30                BST *newBST = new BST(val);
31                right = newBST;
32            } else {
33                right->insert(val);
34            }
35        }
36        return *this;
37    }
38
39    // Average: O(log(n)) time | O(log(n)) space
40    // Worst: O(n) time | O(n) space
41    bool contains(int val) {
42        if (val < value) {
43            if (left == NULL) {
44                return false;
45            } else {
46                return left->contains(val);
47            }
48        } else if (val > value) {
49            if (right == NULL) {
50                return false;
51            } else {
52                return right->contains(val);
53            }
54        } else {
55            return true;
56        }
57    }
58
59    // Average: O(log(n)) time | O(log(n)) space
60    // Worst: O(n) time | O(n) space
61    BST &remove(int val, BST *parent = NULL) {
62        if (val < value) {
63            if (left != NULL) {
64                left->remove(val, this);
65            }
66        } else if (val > value) {
67            if (right != NULL) {
68                right->remove(val, this);
69            }
70        } else {
71            if (left != NULL && right != NULL) {
72                value = right->getMinValue();
73                right->remove(value, this);
74            } else if (parent == NULL) {
75                if (left != NULL) {
76                    value = left->value;
77                    right = left->right;
78                    left = left->left;
79                } else if (right != NULL) {
80                    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}
```

Your Solutions

Run Code

Solution 1

Solution 2

Solution 3

```
1 #include <vector>
2 using namespace std;
3
4 // Do not edit the class below except for
5 // the insert, contains, and remove methods.
6 // Feel free to add new properties and methods
7 // to the class.
8 class BST {
9 public:
10    int value;
11    BST *left;
12    BST *right;
13
14    BST(int val) {
15        value = val;
16        left = NULL;
17        right = NULL;
18    }
19
20    BST &insert(int val) {
21        // Write your code here.
22        // Do not edit the return statement of this method.
23        return *this;
24    }
25
26    bool contains(int val) {
27        // Write your code here.
28        return false;
29    }
30
31    BST &remove(int val) {
32        // Write your code here.
33        // Do not edit the return statement of this method.
34        return *this;
35    }
36 };
37
```

Custom Output

Raw Output

Submit Code

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

Run or submit code when you're ready.