AlgoExpert   Quad Layout   Java   12px   Sublime   Monokai   00:00:0

Our Solution(s)   Run Code

Solution 1   Solution 2

```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

class Program {
  static class BST {
    public int value;
    public BST left;
    public BST right;

    public BST(int value) {
      this.value = value;
    }

    // Average: O(log(n)) time | O(log(n)) space
    // Worst: O(n) time | O(n) space
    public BST insert(int value) {
      if (value < this.value) {
        if (left == null) {
          BST newBST = new BST(value);
          left = newBST;
        } else {
          left.insert(value);
        }
      } else {
        if (right == null) {
          BST newBST = new BST(value);
          right = newBST;
        } else {
          right.insert(value);
        }
      }
      return this;
    }

    // Average: O(log(n)) time | O(log(n)) space
    // Worst: O(n) time | O(n) space
    public boolean contains(int value) {
      if (value < this.value) {
        if (left == null) {
          return false;
        } else {
          return left.contains(value);
        }
      } else if (value > this.value) {
        if (right == null) {
          return false;
        } else {
          return right.contains(value);
        }
      } else {
        return true;
      }
    }

    // Average: O(log(n)) time | O(log(n)) space
    // Worst: O(n) time | O(n) space
    public BST remove(int value) {
      remove(value, null);
      return this;
    }

    public void remove(int value, BST parent) {
      if (value < this.value) {
        if (left != null) {
          left.remove(value, this);
        }
      } else if (value > this.value) {
        if (right != null) {
          right.remove(value, this);
        }
      } else {
        if (left != null && right != null) {
          this.value = right.getMinValue();
          right.remove(this.value, this);
        } else if (parent == null) {
          if (left != null) {
            this.value = left.value;
            right = left.right;
            left = left.left;
          } else if (right != null) {
            this.value = right.value;
            left = right.left;
            right = right.right;
          } else {
            // This is a single-node tree; do nothing.
          }
        } else if (parent.left == this) {
          parent.left = left != null ? left : right;
        } else if (parent.right == this) {
          parent.right = left != null ? left : right;
        }
```

Your Solutions   Run Code

Solution 1   Solution 2   Solution 3

```java
class Program {
  static class BST {
    public int value;
    public BST left;
    public BST right;

    public BST(int value) {
      this.value = value;
    }

    public BST insert(int value) {
      // Write your code here.
      // Do not edit the return statement of this method.
      return this;
    }

    public boolean contains(int value) {
      // Write your code here.
      return false;
    }

    public BST remove(int value) {
      // Write your code here.
      // Do not edit the return statement of this method.
      return this;
    }
  }
}

```

Custom Output   Raw Output   Submit Code

```
      }
    }

    public int getMinValue() {
      if (left == null) {
        return this.value;
      } else {
        return left.getMinValue();
      }
    }
  }
}

```

Run or submit code when you're ready.