AlgoExpert | Quad Layout | JavaScript | 12px | Sublime | Monokai | 00:00:0

Our Solution(s) Run Code

Solution 1 Solution 2

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

class BST {
  constructor(value) {
    this.value = value;
    this.left = null;
    this.right = null;
  }

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

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

  // Average: O(log(n)) time | O(log(n)) space
  // Worst: O(n) time | O(n) space
  remove(value, parent = null) {
    if (value < this.value) {
      if (this.left !== null) {
        this.left.remove(value, this);
      }
    } else if (value > this.value) {
      if (this.right !== null) {
        this.right.remove(value, this);
      }
    } else {
      if (this.left !== null && this.right !== null) {
        this.value = this.right.getMinValue();
        this.right.remove(this.value, this);
      } else if (parent === null) {
        if (this.left !== null) {
          this.value = this.left.value;
          this.right = this.left.right;
          this.left = this.left.left;
        } else if (this.right !== null) {
          this.value = this.right.value;
          this.left = this.right.left;
          this.right = this.right.right;
        } else {
          // This is a single-node tree; do nothing.
        }
      } else if (parent.left === this) {
        parent.left = this.left !== null ? this.left : this.right;
      } else if (parent.right === this) {
        parent.right = this.left !== null ? this.left : this.right;
      }
    }
    return this;
  }

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

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```javascript
// Do not edit the class below except for
// the insert, contains, and remove methods.
// Feel free to add new properties and methods
// to the class.
class BST {
  constructor(value) {
    this.value = value;
    this.left = null;
    this.right = null;
  }

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

  contains(value) {
    // Write your code here.
  }

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

// Do not edit the line below.
exports.BST = BST;
```

Custom Output Raw Output Submit Code

```
  }
}

exports.BST = BST;

```

Run or submit code when you're ready.