**Implementing a Binary Search Tree and Searching a Value in BST**

class Node {

  constructor(data) {

    this.data = data;

    this.left = null;

    this.right = null;

  }

}

class BST {

  constructor() {

    this.root = null;

  }

  insert(val) {

    const newNode = new Node(val);

    if (!this.root) {

      this.root = newNode;

      return;

    }

    let prev = null;

    let current = this.root;

    while (current) {

      if (current.data > val) {

        prev = current;

        current = current.left;

      } else if (current.data < val) {

        prev = current;

        current = current.right;

      }

    }

    if (prev.data > val) {

      prev.left = newNode;

    } else if (prev.data < val) {

      prev.right = newNode;

    }

  }

  printInOrder(node = this.root) {

    if (!node) return null;

    this.printInOrder(node.left);

    console.log(node.data);

    this.printInOrder(node.right);

  }

  search(node, val) {

    if (!node) return false;

    if (node.data === val) {

      return true;

    }

    if (node.data > val) {

      return this.search(node.left, val);

    } else {

      return this.search(node.right, val);

    }

  }

}

const tree = new BST();

tree.insert(5);

tree.insert(3);

tree.insert(7);

tree.insert(2);

tree.insert(4);

tree.insert(6);

tree.insert(8);

tree.printInOrder();

console.log(tree.search(tree.root ,17876))

console.log(tree.search(tree.root ,6))