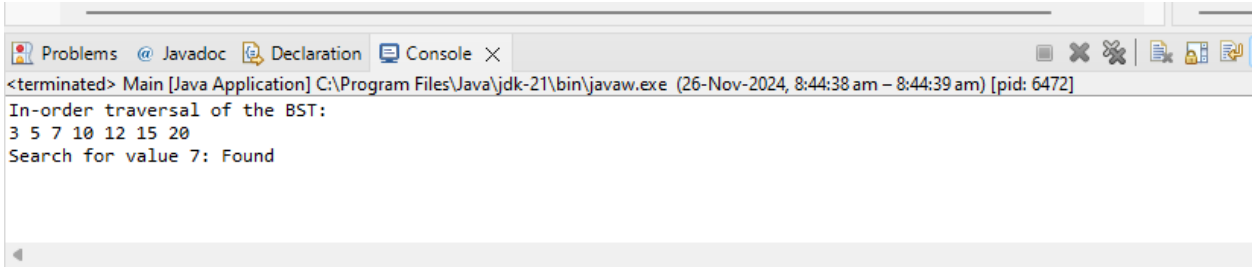
# DSA Coding Practice-9

**1.Binary Search Tree:**

```
class TreeNode {
    int value;
    TreeNode left, right;
        public TreeNode(int value) {
        this.value = value;
        this.left = null;
        this.right = null;
    }
}
class BinarySearchTree {
    TreeNode root;
    public BinarySearchTree() {
        root = null;
    }
    public void insert(int value) {
        root = insertRec(root, value);
    }
    private TreeNode insertRec(TreeNode root, int value) {
            if (root == null) {
            root = new TreeNode(value);
            return root;
        }
        if (value < root.value) {
            root.left = insertRec(root.left, value);
        } else if (value > root.value) {
            root.right = insertRec(root.right, value);
        }
        return root;
    }
    public void inOrderTraversal() {
        inOrderRec(root);
    }
    private void inOrderRec(TreeNode root) {
        if (root != null) {
            inOrderRec(root.left);
            System.out.print(root.value + " ");
            inOrderRec(root.right);
        }
    }
    public boolean search(int value) {
        return searchRec(root, value);
    }
```

```
    private boolean searchRec(TreeNode root, int value) {
        if (root == null || root.value == value) {
            return root != null;
        }
        if (value > root.value) {
            return searchRec(root.right, value);
        }
        return searchRec(root.left, value);
    }
}
public class Main {
    public static void main(String[] args) {
        BinarySearchTree bst = new BinarySearchTree();
        bst.insert(10);
        bst.insert(5);
        bst.insert(15);
        bst.insert(3);
        bst.insert(7);
        bst.insert(12);
        bst.insert(20);
        System.out.println("In-order traversal of the BST:");
        bst.inOrderTraversal();
        System.out.println();
        int searchValue = 7;
        boolean found = bst.search(searchValue);
        System.out.println("Search for value " + searchValue + ": " + (found ? "Found" : "Not
Found"));
    }
}
```

**Output:**



**2.Binary Search Tree:**

```
class TreeNode {
   int value;
   TreeNode left, right;
   public TreeNode(int value) {
      this.value = value;
      this.left = null;
      this.right = null;
   }
}
public class BinaryTree {
   TreeNode root;
   public boolean isBST() {
      return isBSTUtil(root, Integer.MIN_VALUE, Integer.MAX_VALUE);
   }
   private boolean isBSTUtil(TreeNode node, int min, int max) {
      if (node == null) {
         return true;
      }
      if (node.value <= min || node.value >= max) {
         return false;
      }
      return isBSTUtil(node.left, min, node.value) &&
            isBSTUtil(node.right, node.value, max);
   }
   public static void main(String[] args) {
      BinaryTree tree = new BinaryTree();
      tree.root = new TreeNode(10);
      tree.root.left = new TreeNode(5);
      tree.root.right = new TreeNode(15);
      tree.root.left.left = new TreeNode(3);
      tree.root.left.right = new TreeNode(7);
      tree.root.right.left = new TreeNode(12);
      tree.root.right.right = new TreeNode(18);
      if (tree.isBST()) {
         System.out.println("The tree is a Binary Search Tree.");
      } else {
         System.out.println("The tree is NOT a Binary Search Tree.");
      }
      tree.root.right.left = new TreeNode(8);
      if (tree.isBST()) {
         System.out.println("The tree is a Binary Search Tree.");
      } else {
         System.out.println("The tree is NOT a Binary Search Tree.");
```

```
        }
    }
}
```

**Output:**

```
Problems  @ Javadoc  Declaration  Console
<terminated> BinaryTree [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe  (26-Nov-2024, 8:50:29 am – 8:50:30 am) [pid: 8076]
The tree is a Binary Search Tree.
The tree is NOT a Binary Search Tree.
```