1.

/\*\*

\* Returns whether {@code x} is in {@code t}.

\*

\* **@param** <T>

\* type of {@code BinaryTree} labels

\* **@param** t

\* the {@code BinaryTree} to be searched

\* **@param** x

\* the label to be searched for

\* **@return** true if t contains x, false otherwise

\* **@requires** IS\_BST(t)

\* **@ensures** isInTree = (x is in labels(t))

\*/

public static <T extends Comparable<T>> boolean isInTree(BinaryTree<T> t,

T x) {

boolean inTree = false;

BinaryTree<T> left = t.newInstance();

BinaryTree<T> right = t.newInstance();

T node = t.root();

if (t.size() > 1) {

t.disassemble(left, right);

// goto left or right branch

if (node.compareTo(x) == -1) {

inTree = *isInTree*(left, x);

} else if (node.compareTo(x) == 1) {

inTree = *isInTree*(right, x);

}

t.assemble(node, left, right);

}

if (!inTree) {

inTree = node.equals(x);

}

// This line added just to make the component compilable.

return inTree;

}

2.

