**CS1083**

**Assignment #11**

**Daniyal Khan**

**3765942**

**BinarySearchTree.java:**

*public* *class* BinarySearchTree {

*public* BSTNode root;

*public* *BinarySearchTree* () {

root = null;

}

*public* boolean *insert*(Customer c) {

BSTNode newNode = *new* *BSTNode*(c);

*if* (root == null) {

root = newNode;

*return* true;

}

*return* *insertRec*(root, newNode);

}

*private* boolean *insertRec*(BSTNode curr, BSTNode toAdd) {

*if* (toAdd.*data*.*compareTo*(curr.*data*) == 0) {

curr.*frequency* += 1;

*return* false;

} *else* *if* (toAdd.*data*.*compareTo*(curr.*data*) < 0) {

*if* (curr.*left* == null) {

curr.*left* = toAdd;

*return* true;

}

*return* *insertRec*(curr.*left*, toAdd);

} *else* {

*if* (curr.*right* == null) {

curr.*right* = toAdd;

*return* true;

}

*return* *insertRec*(curr.*right*, toAdd);

}

}

*public* int *search*(Customer c) {

BSTNode newNode = *new* *BSTNode*(c);

*if* (root == null) {

*return* 0;

}

*return* *searchRec*(root, newNode);

}

*private* int *searchRec*(BSTNode curr, BSTNode toFind) {

*if* (curr == null) {

*return* 0;

}

*if* (toFind.*data*.*compareTo*(curr.*data*) == 0) {

*return* curr.*frequency*;

} *else* *if* (toFind.*data*.*compareTo*(curr.*data*) < 0) {

*return* *searchRec*(curr.*left*, toFind);

} *else* {

*return* *searchRec*(curr.*right*, toFind);

}

}

*public* void *printInOrder*() {

*if* (root == null) {

*return*;

}

*printInOrderRec*(root);

}

*private* void *printInOrderRec*(BSTNode curr) {

*if* (curr == null) {

*return*;

}

*printInOrderRec*(curr.*left*);

System.*out*.*println*(curr.*data*);

*printInOrderRec*(curr.*right*);

}

*public* void *printPreOrder*() {

*if* (root == null) {

*return*;

}

*printPreOrderRec*(root);

}

*private* void *printPreOrderRec*(BSTNode curr) {

*if* (curr == null) {

*return*;

}

System.*out*.*println*(curr.*data*);

*printPreOrderRec*(curr.*left*);

*printPreOrderRec*(curr.*right*);

}

*public* void *printPostOrder*() {

*if* (root == null) {

*return*;

}

*printPostOrderRec*(root);

}

*private* void *printPostOrderRec*(BSTNode curr) {

*if* (curr == null) {

*return*;

}

*printPostOrderRec*(curr.*left*);

*printPostOrderRec*(curr.*right*);

System.*out*.*println*(curr.*data*);

}

*private* *class* BSTNode {

*public* BSTNode left;

*public* BSTNode right;

*public* Customer data;

*public* int frequency;

*public* *BSTNode*(Customer dataIn) {

this.*left* = null;

this.*right* = null;

this.*data* = dataIn;

frequency = 1;

}

}

}

/\*

*\* Pre Order Traversal: VLR*

*\* In-Order Traversal: LVR*

*\* Post-Order Traversal: LRV*

*\**

*\* L = left*

*\* R = right*

*\* V = visit*

\*/

**BSTDriver.java:**  
  
*public* *class* BSTDriver {

*public* *static* void *main* (String args[]) {

BinarySearchTree bst1 = *new* *BinarySearchTree*();

Customer c1 = *new* *Customer*("Elijah");

Customer c2 = *new* *Customer*("Hosford");

Customer c3 = *new* *Customer*("Gavin");

Customer c4 = *new* *Customer*("Joseph");

Customer c5 = *new* *Customer*("Connor");

Customer c6 = *new* *Customer*("Sarah");

Customer c7 = *new* *Customer*("Brayden");

Customer c8 = *new* *Customer*("Luna");

Customer c9 = *new* *Customer*("Nigel");

// *Not added in the binary search tree*

Customer c12 = *new* *Customer*("Daniel");

bst1.*insert*(c9);

bst1.*insert*(c1);

bst1.*insert*(c8);

bst1.*insert*(c7);

bst1.*insert*(c6);

bst1.*insert*(c2);

bst1.*insert*(c3);

bst1.*insert*(c4);

bst1.*insert*(c5);

bst1.*insert*(c6); // *Duplicates*

bst1.*insert*(c6);

bst1.*insert*(c6);

bst1.*insert*(c1);

System.*out*.*println*("Search for Elijah: " + bst1.*search*(c1));

System.*out*.*println*("Search for Sarah: " + bst1.*search*(c6));

System.*out*.*println*("Search for Gavin: " + bst1.*search*(c3));

System.*out*.*println*("Search for Daniel (does not exist): " + bst1.*search*(c12));

System.*out*.*println*();

System.*out*.*println*("In Order:");

bst1.*printInOrder*();

System.*out*.*println*();

System.*out*.*println*("Pre Order:");

bst1.*printPreOrder*();

System.*out*.*println*();

System.*out*.*println*("Post Order:");

bst1.*printPostOrder*();

System.*out*.*println*();

}

}

**Customer.java:**  
  
  
*public* *class* Customer *implements* Comparable<Customer>{

*private* int uniqueID;

*private* String name;

*private* *static* int id = 0;

*public* *Customer*(String name){

this.*name* = name;

uniqueID = id++;

}

*public* int *compareTo*(Customer other){

*return* this.*uniqueID* - other.*uniqueID*;

}

*public* String *toString*(){

*return* uniqueID + ": " + name;

}

}