**Tree Class**

**public** **class** Tree

{

**private** TNode root;

Tree()

{

root = **null**;

}

**public** **void** insert(**int** data)

{

TNode node = **new** TNode(data);

**if**(root == **null**)

{

root = node;

**return**;

}

insertNode(root, node);

}

**public** **void** insertNode(TNode current, TNode node)

{

**if**(current.getData() > node.getData())

{

**if**(current.left == **null**)

{

current.left = node;

}

**else**

{

insertNode(current.left, node);

}

}

**else** **if**(current.getData() <= node.getData())

{

**if**(current.right == **null**)

{

current.right = node;

}

**else**

{

insertNode(current.right, node);

}

}

}

**public** **boolean** find (**int** data)

{

**return** find\_node(root, data);

}

**private** **boolean** find\_node (TNode cur, **int** data)

{

**if**(cur == **null**)

**return** **false**;

**if**(cur.getData() == data)

**return** **true**;

**if** (cur.getData() > data)

**return** find\_node(cur.left, data);

**else**

**return** find\_node(cur.right, data);

}

**public** **void** traverse()

{

//traverse\_preorder(root);

traverse\_inorder(root);

}

**public** **void** traverse\_preorder (TNode cur)

{

**if**(cur == **null**)

**return**;

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

traverse\_preorder(cur.left);

traverse\_preorder(cur.right);

}

**public** **void** traverse\_inorder(TNode cur)

{

**if**(cur == **null**)

**return**;

**if** (cur.left == **null** && cur.right == **null**)

{

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

}

**else**

{

traverse\_inorder(cur.left);

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

traverse\_inorder(cur.right);

}

}

}

**TNode Class**

**public** **class** TNode

{

**public** TNode left;

**public** TNode right;

**private** **int** data;

TNode(**int** input)

{

data = input;

}

**public** **int** getData()

{

**return** data;

}

}