Trees, recursion and callbacks

CS201
Spring 2005
Week 11
public class TreeNode {
    private TreeNode left;
    private TreeNode right;
    private Object data;

    // let's use Eclipse to generate the constructor, setters and getters

    CS201 – Spring 2005
Let's build a tree by hand

In order traversal:

```java
TreeNode btn = new BinaryTreeNode(
    new Integer(6),
    new BTN(new Integer(2)),
    new BTN(new Integer(9), null, null)
)
```

Here is a picture:

- Visit left
- Print value at node
- Visit right

Start 12:14
Stop 12:19
In-order traversal

- visit the left
- visit the current node
- visit the right node
- BinaryTreeNode btn =
  new BinaryTreeNode(new Integer(6),
  new BinaryTreeNode(new Integer(3),null,null),
  new BinaryTreeNode(new Integer(9),null,null));
btn.inorder() – what is the result if we print out nodes in order?
Inorder() method

public static void inOrder() {
    HOW DO WE WRITE THIS?
    Consider that a binary tree node is recursively defined:
    TreeNode | Object data
    TreeNode left
    TreeNode right

    Try it!  CS201 — Spring 2000
public class BinaryTreeNode {
    private int data;
    private BinaryTreeNode left;
    private BinaryTreeNode right;

    public void inOrder() {
        if (left != null) {
            left.printInOrder();
        }

        System.out.println(data + String.valueOf(data));

        if (right != null) {
            right.printInOrder();
        }
    }

    public int data() {
        return data;
    }
}

CS3201 - Spring 2005

1 3
2
3 1 2 3 6 9
Callbacks

public interface Callback {
    public void visit(Object o);
}

CS201 - Spring 2005
Now, can create a very specific callback routine

```java
public class printcallback implements Callback {
    • include visit methods for types of node
}
```
Let's do some testing and debugging in Eclipse