CS 635 Advanced Object-Oriented Design & Programming
Spring Semester, 2010
Doc 15 Comments on Assignment 2
March 16, 2010

Problem 1

-2 points

BSTNode Add

```
class BSTNode {

    public boolean add(String valueToAdd) {
        if ( valueToAdd < value )
            if (left.isNull())
                return left.add(valueToAdd);
            else
                left = new BSTNode(valueToAdd);
                return true;
        if ( valueToAdd > value )
            if (right.isNull())
                return right.add(valueToAdd);
            else
                right = new BSTNode(valueToAdd);
                return true;
        blah
    }
```


-5 points

BSTNode Add

```
class BSTNode {

    public boolean add(String valueToAdd) {
        if ( valueToAdd < value )
            return left.add(valueToAdd);
        if ( valueToAdd > value )
            return right.add(valueToAdd);
        blah
    }
```

Issues

```
class NullNode extends Node {


    public boolean add(BSTnode nd, String newValue) throws BSTexception {
        if (newValue.compareTo(nd.value) < 0) {
            nd.leftChild = new BSTnode(newValue);
            return true;
        } else {
            nd.rightChild = new BSTnode(newValue);
            return true;
        }
    }
}
```

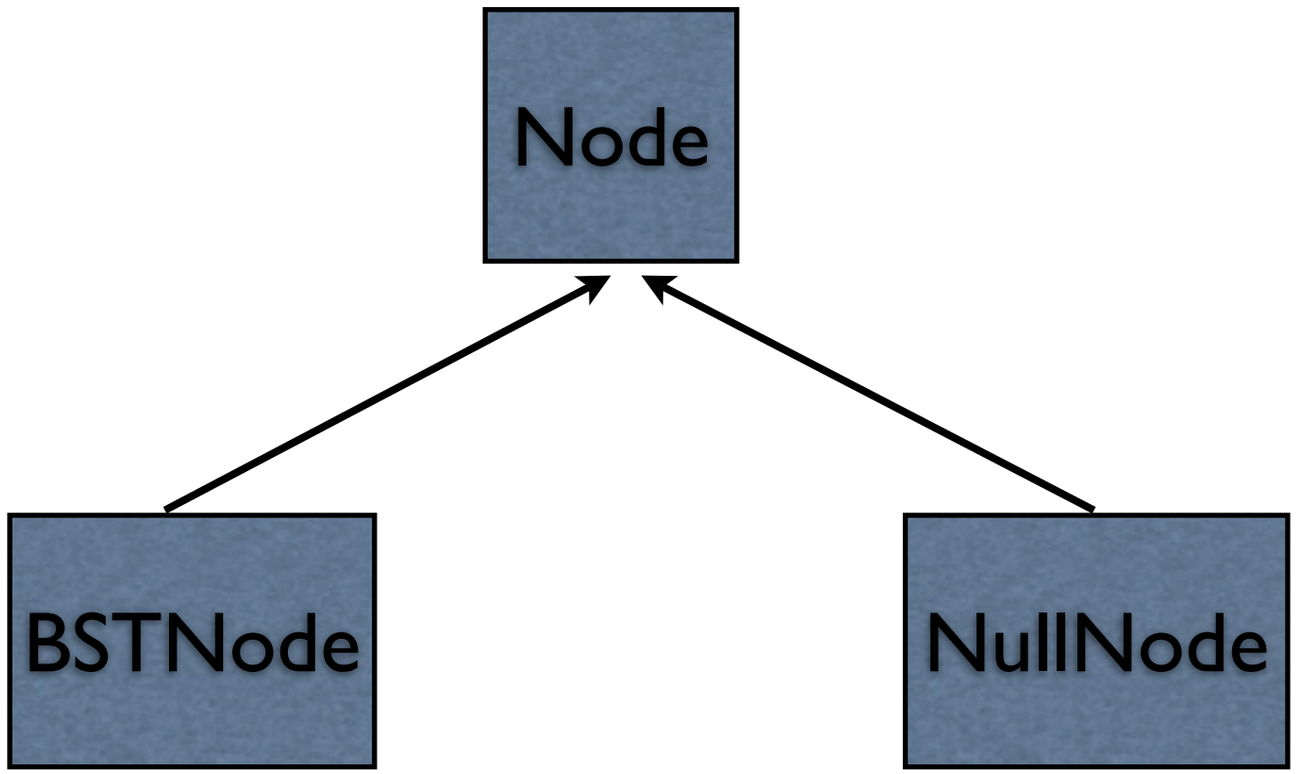
Issues

```
class NullNode extends Node {


    public boolean add(BSTNode parent, String newValue) throws BSTException {
        if (newValue.compareTo(parent.value) < 0) {
            parent.leftChild = new BSTNode(newValue);
            return true;
        } else {
            parent.rightChild = new BSTNode(newValue);
            return true;
        }
    }
}
```

Why is this better?

```
class NullNode extends Node {

    public boolean add(BSTNode parent, String newValue) throws BSTException {
        parent.privateAdd(newValue);
    }
}
```

Should Node Declare left, right fields?



Issues?

```
public class BSTIterator implements Iterator<String> {
    private Stack<BSTNode> stack = new Stack<BSTNode>();

    private void pushElements(Node node) {
        while (node.get() ! = null ) {
            stack.push( (BSTNode) node );
            node = node.getLeft();
        }
    }
}
```

Better?

```
public class BSTIterator implements Iterator<String> {
    private Stack<BSTNode> pathToRoot = new Stack<BSTNode>();

    private void pushLeftPathToLeaf(Node current) {
        while (current.get() ! = null ) {
            pathToRoot.push( (BSTNode) current );
            current = current.getLeft();
        }
    }
}
```

Why 3

```
public class BSTNode extends AbstractCollection<String> {

    public AbstractCollection<String> root = new NullNode();
    public AbstractCollection<String> left = new NullNode();
    public AbstractCollection<String> right = new NullNode();
```