EE360T: Software Testing
Problem Set 4

Out: April 22 2011; **Due: May 1, 2011 11:59pm**
Submission: via blackboard
Maximum points: 40

Representation level input generation

You are to use the Java PathFinder model checker to implement a Korat-like pruning-based search for input generation at the representation level.

Consider the following class declaration for binary trees:

```
public class BinaryTree {
    Node root;
    int size;

    static class Node {
        Node left, right;
    }

    public boolean repOk() {
        // precondition: true
        // postcondition: the input tree is acyclic
        ...
    }
```

(a) [5 points]

Implement the method `repOk`.

(b) [10 points]

Implement the following method `filterBasedGenerator` (in class `BinaryTree`), which takes an input `n`, exhaustively enumerates all (valid or invalid) `BinaryTree` objects that have up to `n` nodes, and uses `repOk` as a filter to output the valid ones:

```
    void filterBasedGenerator(int n) {
        // allocate objects
        ...
```

```
    // set field domains
    ...

    // assign field values non-deterministically
    ...

    // run repOk to check validity and output if valid
    ...
}
```

(c) [20 points]

Hand instrument your code to enable Korat-like pruning-based generation. In particular, your instrumentation should:

- add shadow boolean fields to enable monitoring of field accesses;
- add methods that enable non-deterministic field assignments;
- replace field accesses with appropriate method invocations in `repOk`—you may want to have a separate instrumented version of `repOk` with a different name.

(d) [5 points]

Implement the following method `pruningBasedGenerator` (in class `BinaryTree`), which takes an input `n` and generates all valid trees with up to `n` nodes, using `repOk` to prune the search:

```
void pruningBasedGenerator(int n) {
    // allocate objects
    ...

    // set field domains
    ...

    // run instrumented repOk to enable non-deterministic field
    // assignments on field access, and output if valid
    ...
}
```