EE360T/EE382V: Software Testing
Problem Set 1

Out: Feb 6, 2017; **Due: Feb 13, 2017 11:59pm**
Submission: *.zip via Canvas
Maximum points: 40

1 Testing data structures

Consider the following implementation of a singly-linked list data structure, which represents a container for Fibonacci series:

```
package pset1;

import java.util.HashSet;
import java.util.Set;

public class FibList {
        Node header;
        int size;

        static class Node {
                int elem;
                Node next;
        }

        public FibList() {
                header = n1;
                Node n1 = new Node();
                Node n2 = new Node();
                Node n3 = new Node();
                n1.elem = 1; n1.next = n2;
                n2.elem = 1; n2.next = n3;
                n3.elem = 2;
                size = 3;
        }

        public boolean repOk() {
                // postcondition: returns true iff (1) <this> is an acyclic list, i.e.,
                //                          there is no path from a node to itself;
                //                          (2) the list elements form a Fibonacci series,
                //                          i.e., the element in any node is the sum of
                //                          the elements in the two preceding nodes
                //                          (if they exist); and (3) size >=3

                if (size < 3) return false;
                Set<Node> visited = new HashSet<Node>();
                Node n = header;
                while (n != null) {
                        if (!visited.add(n)) {
                                return false;
                        }
                        if (n.next != null) {
                                if (n.next.next != null) {
```

```
                                        if (n.next.next.elem != n.elem + n.next.elem) {
                                                return false;
                                        }
                                }
                        }
                        n = n.next;
                }
                return size == visited.size();
        }

        public int augment() {
                // precondition: this.repOk()
                // postcondition: adds a new node at the end of the list w.r.t. repOk
                //                and returns the element in this node;
                //                the rest of the list is unmodified;
        }
}
```

1.1 Implementing `augment` [4 points]

Implement the method `augment` as specified.

1.2 Testing `augment` [6 points]

Implement the two test methods in the following class `FibListAugmentTester` as specified:

```
package pset1;

import static org.junit.Assert.*;
import org.junit.Test;

public class FibListAugmentTester {
    @Test public void test0() {
        FibList l = new FibList();
        assertTrue(l.repOk());
        l.augment();

        // write a sequence of assertTrue method invocations that
        // perform checks on the values for all the declared fields
        // of list and node objects reachable from l

        assertTrue(l.header != null);
        // your code goes here

    }

    @Test public void test1() {
        FibList l = new FibList();
        assertTrue(l.repOk());
        l.augment();
        assertTrue(l.repOk());
        l.augment();
        assertTrue(l.repOk());

        // write a sequence of assertTrue method invocations that
        // perform checks on the values for all the declared fields
        // of list and node objects reachable from l

        assertTrue(l.header != null);
        // your code goes here
    }
}
```

1.3 Testing `repOk` [10 points]

Consider testing the method `repOk` by writing a test suites that consists of valid or invalid lists. Specifically, implement test methods in the following class `FibListRepOkTester` such that the test suite achieves full statement coverage of the `repOk` method as given and each test includes at least one test assertion:

```
package pset1;

import static org.junit.Assert.*;
import org.junit.Test;
...

public class FibListRepOkTester {
    @Test public void t0() {
        ...
    }

    @Test public void t1() {
        ...
    }

    ...
}
```

2 Textbook execrises

Solve the following problems from the Software Testing textbook:

1. [6 points] Exercises – Section 2.2.1 Question 5 (Page 43)
2. [7 points] Exercises – Section 2.2.1 Question 6 (Page 43)
3. [7 points] Exercises – Section 2.3 Question 1 (Pages 60–61)