```java
//******************************************************************************
//
// File:    Simulator.java
// Package: ---
// Unit:    Class Simulator
//
//******************************************************************************

import edu.rit.pj2.Loop;
import edu.rit.pj2.Task;
import edu.rit.pj2.vbl.DoubleVbl;
import edu.rit.pj2.vbl.IntVbl;
import edu.rit.util.Random;

/**
 * Class is responsible for the majority of the runtime of the program. It
 * generates the given number of networks in parallel by utilizing Prof. Alan
 * Kaminsky's PJ2 library.
 *
 * @author Jimi Ford (jhf3617)
 * @version 4-4-2015
 */
public class Simulator {

    private Task ref;
    private int v;
    private int trials;
    private long seed;
    private IntVbl.Sum countConnected;
    private DoubleVbl.Mean averagePower;

    /**
     * Construct a Simulator
     * @param ref the reference to the main task - necessary for utilizing the
     *      class's parallelFor method
     * @param v number of space stations (or nodes in the graph)
     * @param trials the number of random networks to generate
     * @param seed seed value for the PRNG used in instantiated classes
     */
    public Simulator(Task ref, int v, int trials, long seed) {
        this.ref = ref;
        this.v = v;
        this.trials = trials;
        this.seed = seed;
        countConnected = new IntVbl.Sum();
        averagePower = new DoubleVbl.Mean();
    }

    /**
     * Run all <TT>trials<TT>
     * @return a SimulationResult containing the findings of the given number of
     *          simulations
     */
    public SimulationResult simulate() {
        ref.parallelFor(0, trials - 1).exec(new Loop() {

            Random prng;
            DoubleVbl.Mean thrAverage;
```

```
59              IntVbl.Sum thrCount;
60
61              // (Non-javadoc)
62              public void start() {
63                  prng = new Random(seed + rank());
64                  thrAverage = threadLocal(averagePower);
65                  thrCount = threadLocal(countConnected);
66              }
67
68              // (Non-javadoc)
69              public void run(int i) throws Exception {
70                  SpaceNetwork sn = new SpaceNetwork(prng, v);
71                  if(sn.isConnected()) {
72                      thrCount.item++;
73                  }
74                  sn.accumulatePower(thrAverage);
75              }
76          });
77          return new SimulationResult(
78                  v,
79                  trials,
80                  countConnected.intValue(),
81                  averagePower.doubleValue());
82      }
83  }
84
```