```
// $Id: observereport.java,v 1.6 2013-08-12 19:26:25-07 - - $

//
// Observers/ reporters problem.
// Observer waits for intervals and counts events happening.
// Reporter periodically reports these events
//

import java.util.*;
import static java.lang.Math.*;
import static java.lang.String.*;
import static java.lang.System.*;

class observereport {
   static long nanostart = nanoTime();

   static void print (String fmt, Object... args) {
      Thread self = Thread.currentThread();
      double millisec = (nanoTime() - nanostart) / 1e6;
      String ident = format ("%10.3f: %s(%d): ",
                             millisec, self.getName(), self.getId());
      String message = format (fmt, args);
      out.println (ident + message);
      out.flush();
   }

   static class counter {
      private int count = 0;
      public boolean stop = false;
      synchronized void click () {
         ++count;
      }
      synchronized int reset () {
         int result = count;
         count = 0;
         return result;
      }
   }

   public static void main (String[] args) {
      print ("starting");
      counter counted = new counter ();
      Thread obs = new Thread (new observer (counted, 10));
      Thread rep = new Thread (new reporter (counted, 1000, 10));
      obs.start ();
      rep.start ();
      print ("finished");
   }


   static class observer implements Runnable {
      counter counted;
      long delay_msec;
      Random random = new Random ();
      observer (counter _counted, long _delay_msec) {
         counted = _counted;
         delay_msec = _delay_msec;
      }
      public void run () {
         Thread self = Thread.currentThread();
         self.setName ("observer");
         print ("starting");
         while (! counted.stop) {
            try {
               long delay = abs ((long) (delay_msec
                          + delay_msec * random.nextGaussian ()));
               Thread.sleep (delay);
            }catch (InterruptedException exn) {
               print ("observer interrupted: %s", exn);
            }
            counted.click ();
         }
         print ("finished");
      }
   }

   static class reporter implements Runnable {
      counter counted;
      long delay_msec;
      int cycles;
      reporter (counter countinit, long _delay_msec, int _cycles) {
         counted = countinit;
         delay_msec = _delay_msec;
         cycles = _cycles;
      }
      public void run () {
         Thread self = Thread.currentThread();
         self.setName ("reporter");
         print ("starting");
         for (int count = 0; count < cycles; ++count) {
            try {
               Thread.sleep (delay_msec);
            }catch (InterruptedException exn) {
               print ("reporter interrupted: %s", exn);
            }
            int events = counted.reset ();
            print ("report %3d: events: %5d", count, events);
         }
         counted.stop = true;
         print ("finished");
      }
   }
}

//TEST// runprog -x observereport.lis observereport
//TEST// mkpspdf observereport.ps observereport.java* observereport.lis

```

```
* @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting observereport.java
* observereport.java: $Id: observereport.java,v 1.6 2013-08-12
19:26:25-07 - - $
* javac -Xlint observereport.java
* jar cfm observereport Manifest observereport$counter.class
observereport$observer.class observereport$reporter.class
observereport.class
* chmod +x observereport
* rm -f observereport$counter.class observereport$observer.class
observereport$reporter.class observereport.class
* @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished observereport.java
```

```

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
log: observereport.log
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

     1  Script  : /afs/cats.ucsc.edu/courses/cmps012b-wm/bin/runprog
     2  limit c :    0 max core file size (KB)
     3  limit f : 4194303 max output file size (KB)
     4  limit t : 4294967295 max CPU time (sec)
     5  stdin   : /dev/null
     6  stdout  : observereport.out
     7  stderr  : observereport.err
     8  log     : observereport.log
     9  listing : observereport.lis
    10  Command : observereport
    11  starting: pid 9657: 19:33:43.00
    12  finished: pid 9657: 19:33:53.00, real 10.00, user 0.08, sys 0.02
    13  pstatus: 0x0000 EXIT STATUS = 0

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
stdin: /dev/null
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
stdout: observereport.out
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

     1       0.048: main(1): starting
     2       7.478: main(1): finished
     3       7.786: observer(8): starting
     4       7.834: reporter(9): starting
     5    1009.575: reporter(9): report   0: events:    90
     6    2010.103: reporter(9): report   1: events:    83
     7    3010.645: reporter(9): report   2: events:   110
     8    4011.175: reporter(9): report   3: events:    80
     9    5011.707: reporter(9): report   4: events:    80
    10    6012.225: reporter(9): report   5: events:    95
    11    7012.756: reporter(9): report   6: events:    82
    12    8013.281: reporter(9): report   7: events:    87
    13    9013.815: reporter(9): report   8: events:    94
    14   10014.339: reporter(9): report   9: events:    95
    15   10014.810: reporter(9): finished
    16   10016.828: observer(8): finished

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
stderr: observereport.err
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

```