```
package simcode;
import javasim.*;
import streams.*;

/**
 * @author Michael Kepple
 * @version 22 Mar 2013
 * @see JavaSim, Streams
 */
public class ThreeTeller
{
        static final int SECINSHIFT = 21600;
        static final int BEGINSIM = 0;
        static final int ARRIVAL = 1;
        static final int DEPART = 2;
        static final int ENDSIM = 3;

        public static void main(String[] args)
        {
                new ThreeTeller();
        }

        public ThreeTeller()
        {
                boolean run = true;
                double arrive = 0, total = 0;
                double minimum = Double.MAX_VALUE;
                double maximum = Double.MIN_VALUE;
                ExponentialStream arrTime = new ExponentialStream(60);
                ExponentialStream serTime = new ExponentialStream(150);
                Entity endShift, startShift, retrieved;
                Queue tellerOne, tellerTwo, tellerThree;
                tellerOne = tellerTwo = tellerThree = new Queue();
                timeTable table = new timeTable();
                int clock = 0, entityId = 0, eventRetrieved = 0, customersServed =
0, arrivals = 0;
                startShift = new Entity(entityId, "Start of Shift");
                startShift.event = 0;
                while (clock < SECINSHIFT)
                {
                        try
                        {
                                arrive = arrTime.getNumber();
                        }
                        catch (Exception ex)
                        {
                                ex.printStackTrace();
                        }
                        clock += arrive;
                        entityId++;
                        Entity custArrive = new Entity(entityId, "Customer Arrival"
);
                        custArrive.event = ARRIVAL;
                        table.setim(custArrive, clock);
                }
                endShift = new Entity(entityId, "End of Shift");
                endShift.event = ENDSIM;
                table.setim(endShift, clock);

                while (run)
                {
                        retrieved = table.scan();
                        eventRetrieved = retrieved.event;
```

```
                        switch (eventRetrieved)
                        {
                        case BEGINSIM:
                                System.out.println("Bank simulation has begun");
                                break;
                        case ARRIVAL:
                                System.out.println("Customer arrived at: " + retrie
    ved.time);
                                arrivals++;
                                // Decide which Queue to go in
                                if (tellerOne.size() <= Math.min(tellerTwo.size(),
    tellerThree.size()))
                                        tellerOne.add(retrieved);
                                else if (tellerTwo.size() <= Math.min(tellerOne.siz
    e(), tellerThree.size()))
                                        tellerTwo.add(retrieved);
                                else
                                        tellerThree.add(retrieved);
                                // Determine service time
                                int serviceTime = 0;
                                try
                                {
                                        serviceTime = (int)serTime.getNumber();
                                        total += serviceTime;
                                        if (serviceTime > maximum)
                                                maximum = serviceTime;
                                        else if (serviceTime < minimum)
                                                minimum = serviceTime;
                                }
                                catch (Exception ex)
                                {
                                        ex.printStackTrace();
                                }
                                // Set their departure
                                retrieved.event = 2;
                                table.setim(retrieved, (int)(retrieved.time+service
    Time));
                                break;
                        case DEPART:
                                tellerOne.remove(retrieved);
                                tellerTwo.remove(retrieved);
                                tellerThree.remove(retrieved);
                                customersServed++;
                                System.out.println("Customer left at: " + retrieved
    .time);
                                break;
                        case ENDSIM:
                                System.out.println("Hour is up - Simulation has end
    ed");
                                tellerOne.clear();
                                tellerTwo.clear();
                                tellerThree.clear();
                                run = false;
                        }
                }
                System.out.println("Customers Served: " + customersServed);
                System.out.println("Minimum Service Time: " + minimum);
                System.out.println("Maximum Service Time: " + maximum);
                System.out.println("Average Service Time: " + total/arrivals);
        }
```

```
    }
```