METROPOLITAN STATE UNIVERSITY

ICS 240 - 02: Introduction to Data Structures
Fall 2018

Assignment 6: Airport Simulation using Queues
Out: Friday, November, 2nd, 2018
Due: Friday, November, 16th, 2018

Total points: 50

PROBLEM DESCRIPTION

In this assignment, you are asked to implement an airport simulator to simulate the following scenario:

"A small airport that has only one runway. There will be a queue of planes waiting to land and a queue of planes waiting to take off. However, only one plane can use the runway at a time, so there can be only one takeoff or one landing in progress at any one time. Assume that all landings take the same amount of time, but this need not be the same as the takeoff time. Assume that planes arrive for landing at random times but with a specified probability of a plane arriving during any given minute. Similarly, assume that planes arrive at the takeoff queue at random times but with a (possibly different) specified probability of a departure. (Despite the fact that takeoffs and landings are scheduled, delays make this a reasonable assumption.) Since it is more expensive and more dangerous to keep a plane waiting to land than it is to keep a plane waiting to take off, landings will have priority over takeoffs. Thus, as long as some plane is waiting to land, no plane can take off. Use a clock that is an integer variable that counts the number of minutes simulated. Use a random number generator to simulate arrival and departure times of airplanes."

IMPLEMENTATION

You can implement the airport simulator by extending the car wash simulation as follows:

1. The `Averager` and `BooleanSource` classes are used without any changes.
2. The `Washer` class needs to be replaced by a slightly different class, called `Runway`. The runway class is different than the washer class as follows:
   a. Replace the `secondsForWash` instance variable with two instance variables, `minutesForLanding` and `minutesForTakeoff`. Change the constructor accordingly.

b. The `startService` method needs to take an input parameter to specify whether the service is landing or taking off. Assume landing is type 1 and takeoff is type 0. The method then sets the `serviceTimeLeft` according to the type of the service.

3. The `simulate` method is changed as follows:
   a. Takes five input parameters to represent: `landingTime`, `takeoffTime`, `landingProb`, `takeoffProb`, and `totalTime`.
   b. There should be two instances of `BooleanSource` to simulate landing planes arrival and taking off planes arrival. Similarly, there should be two queues to landing and taking off planes. Also, there should be two instances of `Averager` to calculate the average wait time for landing and average wait time for taking off.

After implementing the simulator as explained above, perform the following experiments:

(5 Points) Experiment 1: Getting Started:

- Run a simulation with the following values

Arrival rate	0.45
Departure rate	0.2
Time to land	2
Time to takeoff	3
Simulation time	600

- Answer these questions

a. How many planes landed?
b. How many planes took off?
c. How long did a plane wait on average to land?
d. How long did a plane wait on average to takeoff?
e. At the end of the simulated time, how many planes were left in the landing queue?
f. At the end of the simulated time, how many planes were left in the taking off queue?

(10 Points) Experiment 2: Arrival time experiment:

a. Holding the departure rate, time to land, time to takeoff, and simulation time constant, run a series of simulations in which the arrival rate varies. You may choose any values you wish for the departure rate, time to land, time to takeoff, and simulation time but be sure to hold those values constant for this experiment.
b. Perform at least 10 simulations with different arrival rates and record the results.

c. Draw a graph showing what happens to the number of planes taking off and the number of planes departing as the arrival rate varies. The number of takeoffs and the number of landings should be shown on the Y-axis (vertical axis) and the arrival rate should use the X-axis (horizontal axis).

d. Draw another graph showing what happens to the wait time for planes taking off and the wait time for planes departing as the arrival rate varies. The wait time should be shown on the Y-axis (vertical axis) and the arrival rate should use the X-axis (horizontal axis).

e. Provide titles for the chart and label the axes and the units clearly.

f. What conclusions can you draw about the behavior of the airport relative to landings and takeoffs as a result of this experiment?

(15 Points) Experiment 3: Departure rate experiment:

a. Holding the arrival rate, time to land, time to takeoff, and simulation time constant, run a series of simulations in which the departure rate varies. Choose the same values for time to land, time to takeoff, and simulation time as you chose in the previous experiment.

b. Perform at least 10 simulations with different departure rates and record the results.

c. Draw a graph showing what happens to the number of planes taking off and the number of planes departing as the arrival rate varies. The number of takeoffs and the number of landings should be shown on the Y-axis (vertical axis) and the departure rate should use the X-axis (horizontal axis).

d. Draw another graph showing what happens to the wait time for planes taking off and the wait time for planes departing as the departure rate varies. The wait time should be shown on the Y-axis (vertical axis) and the departure rate should use the X-axis (horizontal axis).

e. Provide titles for the chart and label the axes and the units clearly.

f. What conclusions can you draw about the behavior of the airport relative to landings, takeoffs as a result of this experiment?

(15 Points) Experiment 4: Number of runways experiment:

In this experiment, you are going to test the effect of extending the airport by adding more runways. You can simulate multiple runways by implementing an array for runways as follows. For example, if you want to simulate 3 runways:

```
RunWay[] runwayArr = new RunWay[3];
for (int i=0 ; i < runwayArr.length; i++)
        runwayArr[i] = new RunWay(landingTime,takeoffTime);
```

Inside the simulation loop, you need to decrement the service time for each run way as follows:

```
for (int i=0; i < runwayArr.length; i++)
                    runwayArr[i].reduceRemainingTime();
```

Also, you need to check if any runway is not busy and assign an airplane to that runway. After modifying your simulation program to accommodate several runways, perform the following experiment.

a. Holding the arrival rate, departure rate, time to land, time to takeoff, and simulation time constant, run a series of simulations in which the number of runways varies. Choose the same values for time to land, time to takeoff, and simulation time as you chose in the previous experiments. Choose high values for landing arrival rate and departure rate so that effect of adding more runways is visible.
b. Perform 5 simulations with the number of airways are set to 1,2,3,4, and 5.
c. Draw a graph showing what happens to the number of planes taking off and the number of planes departing as the number of airways varies. The number of takeoffs and the number of landings should be shown on the Y-axis (vertical axis) and the number of airways should use the X-axis (horizontal axis).
d. Draw another graph showing what happens to the wait time for planes taking off and the wait time for planes departing as the number of runways varies. The wait time should be shown on the Y-axis (vertical axis) and the number of runways should use the X-axis (horizontal axis).
e. Provide titles for the chart and label the axes and the units clearly.
f. What conclusions can you draw about the behavior of the airport relative to landings, takeoffs as a result of this experiment?

WHAT TO SUBMIT?

A word document: Include all the graphs and answers to questions above in a word document.

Electronic: Upload to D2L a zip file that includes the following three files: `Runway.Java`, `AirportSimulatorOneRunway.java`, and `AirportSimulatorMultipleRunways.java`.