HOMEWORK - SPRING 2022

HOMEWORK 4 - due Tuesday, March 22nd no later than 7:00pm

REMINDERS:

- **Be sure your code follows the coding style for CSE214.**
- **Make sure you read the warnings about academic dishonesty. *Remember, all work you submit for homework assignments MUST be entirely your own work. Also, group efforts are not allowed.***
- **Login to your grading account and click "Submit Assignment" to upload and submit your assignment.**
- **You are allowed to use any built-in Java API Data Structure classes to implement this assignment except where noted.**
- **You may use Scanner, InputStreamReader, or any other class that you wish for keyboard input.**

The L. I. double R. has realized that it can leverage its reputation as the absolute worst commuter railroad in America in order to extract even more undeserved money from its passengers. Following models from European trains, they've decided to change all their cars to have two classes - first and second class. They would like to know how their morning commute will be affected by this - what the average wait time will be for first and second class will be (and how many people won't get on the train at all) *there will be no standing passengers*. You will model a LIRR morning service with the following stations: Mineola, Hicksville, Syosset, and Huntington (passengers will only get off at

Jamaica and Penn, so we won't worry about those). Each station will have a probability for a first class and a second class passenger arrival at any given minute. If the first class is full, the first class passengers can go to second class, but not vice versa. Trains will start at Huntington every 5 minutes, and take 5 minutes between stations (this is an idealized world where trains go fast and come often, we can multiply this by a large constant to get realistic data at a later date), and at the end of the simulation, you will print what the average wait time was for passengers at each given station for each given class, and how many passengers were served. We will pretend that there is only one train car, as we can easily multiply this by some constant to get an actual train, but it will make input and output more manageable. We will also pretend that the train is at the station for only one minute. We will also take input for a time that passengers stop showing up for the morning commute.

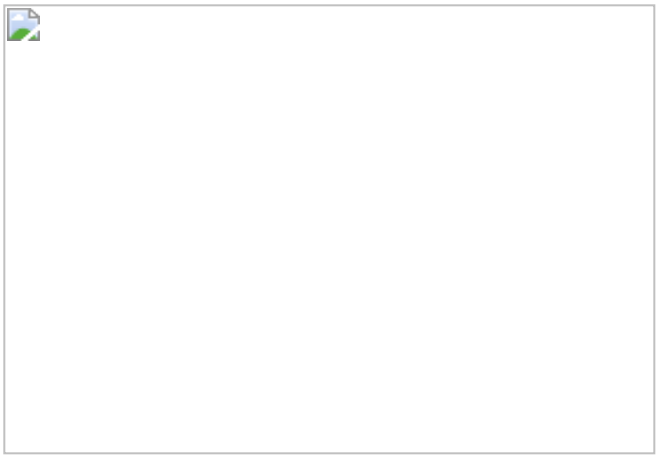
Inputs:

- Train capacity 1st and 2nd class
- Probabilities for each station.
- Time passengers stop showing up (because the LIRR has lower capacity than demand most of the time)

NOTE: All exceptions explicitly thrown in Required Classes except for IllegalArgumentException are custom exceptions that need to be made by you.

UML

The UML Diagram for all the classes specified below is as follows:



Required Classes

Passenger

Write a fully-documented class named Passenger which contains an integer id, the arrival time, and a boolean for isFirstClass. This class must contain methods that allow the user to manipulate the instance variables (getters, setters, etc.)

- int id
- int arrivalTime
- boolean isFirstClass

PassengerQueue

Write a fully documented class called PassengerQueue that has all the regular queue methods. You may create your own class using an array or a liked list, or extend a class implementing the List<T> interface (ie: ArrayList or LinkedList). If you choose to use Java API classes, you **must** extend. You may choose to have an EmptyQueue exception if you wish.

- public void enqueue(Passenger p)
- public Passenger dequeue()
- public String toString()
- public Passenger peek()
- public boolean isEmpty()

BooleanSource

You may copy this from the lecture slides. Basically an object is constructed with a probability p between 0 and 1 and then the occurs() method returns true p*100% of the time.

Station
Write a fully documented class that simulates a station. It must contain two queues, two boolean source instances - one to generate arrivals for first class, and one to generate arrivals for second class.

- public void simulateTimestep()
  - Handles arrival of passengers
- private PassengerQueue firstClass;
- private PassengerQueue secondClass;
- private BooleanSource firstArrival
- private BooleanSource secondArrival

Train
Write a fully documented class called Train to simulate trains. It will contain a queue of the stations the train has to visit (at time%5==0, a new station is dequeued, the list of stations is initialized to contain all the stations on the line, in order). It will also contain a method to simulate station arrival, where passengers are dequeued.

- public void simulateTimeStep()
  - Simulates the passing of one timestep, and dequeues passengers from stations when appropriate
- private int firstCapacity
- private int secondCapacity
- private int time until next Arrival

LIRRSimulator
Write a fully documented class called LIRR simulator. It should have some sort of list of stations, some list of trains currently on the line, and some global variables for number of passengers served, and other useful statistics (see sample I/O for expected end-of-run statistics).

Note on Exceptions: all exceptions should be handled gracefully - they should be caught in the main, and the user should be notified by a nice printout. Your messages will not be graded for creativity, but they should clearly indicate what the problem is (bad index, full list, negative number, etc.). The program should continue to run normally after an exception is encountered. We will not be checking Input Mismatch cases.

Pretty Printing: Here is a tutorial on how to print tables neatly using printf in java. It is highly encouraged that you print the output neatly, as it makes grading much easier.
https://docs.oracle.com/javase/tutorial/java/data/numberformat.html Here's a friendly reference page:
https://alvinalexander.com/programming/printf-format-cheat-sheet

---

General Recommendations

You might want to use the toString() method for CommandStack to make debugging and printing easier. You do not have to do this, but it will help you.

You can feel free to add extra methods and variables if you need.

Program Sample

```
Welcome to the LIRR Simulator, Leaving Irate Riders Regularly
//This sample I/O is supposed to show you the mechanics of the program, the numbers are
made up and may not add up.
Mineola:
Please enter first class arrival probability: .3
Please enter second class arrival probability: .5

Hicksville:
Please enter first class arrival probability: .3
Please enter second class arrival probability: .7

Syosset:
Please enter first class arrival probability: .4
Please enter second class arrival probability: .6

Huntington:
Please enter first class arrival probability: .2
Please enter second class arrival probability: .5

Please enter first class capacity: 7

Please enter second class capacity: 15

Please enter number of trains: 4

Please enter last arrival time of passengers: 14 //Throw exception if after last
station arrival of last train

Begin Simulation:
-------------------------------------------
Time 0:

Station overview:

Mineola:
No first class passenger arrives
Second class passenger ID 1 arrives
Queues:
First [empty]
Second [P1@T0 //Basically Passenger 1 arrived at Time 0
```

```
Hicksville:
First class passenger ID 2 arrives
Second class passenger ID 3 arrives
Queues:
First [P2@T0
Second [P3@T0

Syosset:
No first class passenger arrives
No second class passenger arrives
Queues:
First [empty]
Second [empty]

Huntington:
First class passenger ID 4 arrives
Second class passenger ID 5 arrives
Queues:
First [P4@T0
Second [P5@T0

Trains:

Train 1 arrives at Huntington, There are 0 passengers in first class and 0 in second
class.
Passengers embarking in first class: P4
Passengers embarking in second class: P5

Train 2 will arrive at Huntington in 5 minutes.

Train 3 will arrive at Huntington in 10 minutes.

Train 4 will arrive at Huntington in 15 minutes.

-------

Time 1:

Station overview:

Mineola:
First Class Passenger ID 6 arrives
Second class passenger ID 7 arrives
Queues:
```

```
First [P6@T1
Second [P1@T0,P7@T1

Hicksville:
First class passenger ID 8 arrives
Second class passenger ID 9 arrives
Queues:
First [P2@T0,P8@T1
Second [P3@T0,P9@T1

Syosset:
First class passenger ID 10 arrives
Second class passenger ID 11 arrives
Queues:
First [P10@T1
Second [P11@T1

Huntington:
No first class passenger arrives
No second class passenger arrives
Queues:
First [empty]
Second [empty]

Trains:

Train 1 will arrive in Syosset in 4 minutes

Train 2 will arrive at Huntington in 4 minutes.

Train 3 will arrive at Huntington in 9 minutes.

Train 4 will arrive at Huntington in 14 minutes.


...//Much Later; we skipped a bunch of time steps
-------

Time 15: //No more arrivals

Station overview:

Mineola:
No first class passenger arrives
No second class passenger arrives
```

```
Queues:
First [P6@T1, P14@T3, P18@T4, P23@T6, P28@T7, 40@T10, P45@T11, P47@T12, P55@T14
Second [P1@T0,P7@T1 P15@T3, P19@T4, P29@T7, P33@T9, P48@t12, P50@T13, P56@T14
//The lines here are expected to be very long before first train arrives
Hicksville:
No first class passenger arrives
No second class passenger arrives
Queues:
First [P46@T11, P57@T14
Second [P52@T13, P58@T14

Syosset:
No first class passenger arrives
No second class passenger arrives
Queues:
First [P52@T13, P59@T14
Second [P49@T12, P60@T14

Huntington:
No first class passenger arrives
No second class passenger arrives
Queues:
First [P53@T13, P61@T14
Second [P54@T13, P62@T14

Trains:

Train 1 arrives at Mineola, there are 7 passengers in first class and 10 in second
class.
Passengers embarking in first class: none //it's full
Passengers embarking in second class: P6, P14, P18, P23, P28 //first class has
priority, fill the train, 2nd class will wait

Train 2 arrives at Hicksville, there are 3 passengers in first class, and 5 in second
class.
Passengers embarking in first class: P46, P57
Passengers embarking in second class: P49, P60

Train 3 arrives at Syosset, there are 1 passengers in first class and 2 in second
class.
Passengers embarking in first class: P52, P59
Passengers embarking in second class: P49, P60

Train 4 arrives at Huntington, there are 0 passengers in first class and 0 in second
```

class.
Passengers embarking in first class: P53, P61
Passengers embarking in second class: P54, P62

...//Much Later; we skipped a bunch of time steps

//After a train has passed Mineola, write "Train X has stopped picking up passengers."
//These are sample stats and may not reflect the above simulation. Average wait time may be rounded to the nearest integer.
//Only consider passengers picked up by trains for average wait time. Count first class passengers sitting in second class as 1st class for stats
---

At the end of the simulation:

A total of 63 passengers were served, 5 first class passengers were left without a seat, 9 second class passengers were left without a seat.

At Mineola 8 first class passengers were served with an average wait time of 19 min, 8 second class passengers were served with an average wait time of 23 min. 0 first class passengers and 0 second class passengers were left without a seat.

At Mineola 7 first class passengers were served with an average wait time of 13 min, 12 second class passengers were served with an average wait time of 16 min. 1 first class passengers and 0 second class passengers were left without a seat.

At Syosset 3 first class passengers were served with an average wait time of 8 min, 5 second class passengers were served with an average wait time of 8 min. 2 first class passengers and 3 second class passengers were left without a seat.

At Huntington 8 first class passengers were served with an average wait time of 1 min, 12 second class passengers were served with an average wait time of 2 min. 2 first class passengers and 6 second class passengers were left without a seat.//It's possible for passengers to arrive after the last train passes their station everywhere but the last station.

Course Info | Schedule | Sections | Announcements | Homework | Exams | Help/FAQ | Grades | HOME