CS212 Project 1

Simplified Airline Reservation System

The goal of this project is to create a simple simulation of a booking system for a small airline that runs commuter flights between two cities on a daily basis. Your projects should just keep track of flights for the month of October 2017.

The classes that you develop could be used to create much more complicated simulations involving multiple airlines and many cities, but for this project only a very simple simulation is required.

Your project should use implementations for 5 classes called: *Main*, *Passenger*, *Ticket*, *Airline*, *Flight*.

The *Main* class contains only static methods (and no instance variables). Its static methods are used to run the simulation and call on the other classes as needed. This should be the last class implemented and should carry out the simulation described below. It is best to start with the other four classes. For each of the classes you should begin by deciding what type to use for each of the instance variables. The next few paragraphs outline possible instance variables and methods for these classes. It is possible to implement the simulation using only these variables and methods, however you may use any other choice of variables and methods that allows you to carry out the required simulation.

The class *Airline* contains information about all the flights for an airline. It uses instance variables *name* and *flights*. The variable *flights* should be an *ArrayList* of *Flight* objects. The variable *name* is a *String* that gives the name of the airline.

The class *Passenger* contains information about a passenger. It uses instance variables *firstName*, *lastName*, *address*, *phone* and *myTickets*. The field *myTickets* is an *ArrayList* of tickets that is used to store all the airline tickets that have been booked by a passenger. The other fields should all be stored as Stings.

The class *Ticket* contains information about a ticket. It uses instance variables *ticketNumber*, *myAirline*, *myPassenger*, *myFlight* and *price*. The class should also have a *static* data field called *counter* which stores an integer that is increased every time a new ticket is issued. This static field can be used to make sure that ticket numbers are unique (and not shared by two different tickets). Ticket numbers are integers and prices have type *double*.

The class *Flight* contains information about a flight. It uses instance variables *flightNumber*, *seats*, *filledSeats*, *flightLength*, *airline*, *date*, *originAirport*, *destination*, *departureTime* and *tickets*. The field *tickets* should be an *ArrayList* of tickets that have been issued for the

flight. A *static* field called *counter* should be used to keep account of the different flights and make sure that each gets a different flight number.

For each of these four classes, there is one constructor. The parameters for this constructor should give values for instance variables. However, if the value for an instance variable can be automatically generated no corresponding parameter should be used. So for example the Flight constructor would not use a parameter to specify *filledSeats* since this always begins with the value 0. Similarly, the constructor for a *Ticket* would not have a parameter to specify a *ticketNumber* since this is automatically generated by using the static field counter.

For each instance variable in each class you should implement a getter method. For each instance variable that could be changed you should implement a setter method too. However, for instance variables like a passenger name which are never changed do not supply a setter. This ensures that passenger names can't be accidentally corrupted. For each of the four classes supply a *toString* method that can be used for testing purposes and can also provide useful output in the main simulation.

Several methods are used to find a convenient flight that matches a choice of date and time. It is unlikely that the time that a passenger chooses here will happen to match the exact time of a scheduled flight, accordingly these methods will return all flights that depart on the correct date within 4 hours of the preferred time. The methods that belong to the four classes have the following title lines and tasks.

For the class *Passenger*:

```
void cancel(Ticket t)
  cancels the passenger's ticket t

ArrayList<Flight> findFlights(Airline a, String date, double time, String from)
  finds all flights for an airline on a particular date within 4 hours of a
  particular departure time from a particular city.

Ticket bookFlight(Flight f)
  books a ticket for a particular flight (for the passenger).

boolean holdsTicket(Ticket t)
  reports where the passenger holds a particular ticket
```

For the class *Ticket*:

```
void cancel()
   cancels the ticket
```

For the class *Airline*:

```
void cancel(Ticket t)
  cancels a ticket
void issueRefund(Ticket t)
  issues a refund --- it just prints a message to the screen about which
                      passenger has been credited how much money since in
      this simulation we will not keep track of
                      bank balances for passengers or airlines.
ArrayList<Flight> findFlights(String date, double time, String origin)
  finds all flights for a 4 hour departure window
Ticket book(Passenger p, Flight f)
  books a passenger on a flight
double cost(Flight f)
  gives the cost of a ticket for a particular flight.  Devise you own sensible pricing
  policy so that tickets get more expensive as a flight fills up.
void createFlight(double time, int numSeats, String from, String to)
  creates a new flight for the Airline and makes sure that this flight operates
  every day.
```

For the class *Flight*:

```
boolean matches(String d, double t, String from)
   Does the flight match date d, time t and originAirport from
   to within a 4 hour departure window
boolean hasSpace()
   Are there any empty seats left?
void addTicket(Ticket t)
   Add a newly issued ticket to the flight
boolean holdsTicket(Ticket ticket)
   Does the flight already hold a particular ticket?
void remove(Ticket ticket)
   Remove a canceled ticket from the flight.
double getCost()
   Use the flight's airline's method to generate the cost of the next ticket
   for this flight.
```

For the class *Main*

You should create a main method that operates the following simulation of a user's booking experience. Before the user begins booking flights at step 4, the simulation needs to create an Airline and its scheduled list of flights. It then creates existing bookings so that some flights will already be full or near to full to simulate the possibility that our desired flight is already sold out. These existing bookings are created at step 3. Do not worry about generating sensible names or other data for the passengers at step 3, their only purpose is to

fill seats. The interaction with the user takes place at steps 4 to 7. The user might decide to book several flights here and cancel some of them.

This main method might usefully call on other static methods to carry out parts of its task.

The simulation to be carried out by the main class is as follows:

```
1.  Create a new Airline object
2.  Randomly generate flights for the airline for the month of October 2017.
    There should be about one flight per hour in each direction between 6am and 10pm.
    The same flights should operate each day of the month.
    Flight numbers should be the same from day to day.
3.  Generate 10000 random passengers each of whom books one flight.

4.  Ask the user to create their passenger data.
5.  Run a loop of user interactions to allow the user to book and cancel flights.
6.  Each interaction either quits the loop, books a flight, or cancels a ticket.

7.  When the user finishes, print a list of all of the user's booked tickets.
```

Here is a typical interaction of the program with a user.

```
Ready to book your flights.  Enter your first and last name please:
Donald Duck
Type your address on one line please:
Kissena Blvd, Flushing, NY 11366
Type your phone number on one line please:
718 997 3500
Ready to book your flights between Kennedy and Laguardia for October 2017
Do you want to book or cancel a flight? Answer Yes of No:
Yes
Enter C to cancel, K for a flight from Kennedy, or L for a flight from Laguardia
K
Enter the day in October that you want to fly:
4
Enter an hour you would like to fly (in range 1 - 24)
11
Here are available flights:
Air Queens 102 10.4.17 7.51 from Kennedy to Laguardia ticket cost 130.0
Air Queens 104 10.4.17 8.05 from Kennedy to Laguardia ticket cost 150.0
Air Queens 106 10.4.17 9.18 from Kennedy to Laguardia ticket cost 160.0
Air Queens 108 10.4.17 10.42 from Kennedy to Laguardia ticket cost 190.0
Air Queens 110 10.4.17 11.33 from Kennedy to Laguardia ticket cost 250.0
Air Queens 112 10.4.17 12.08 from Kennedy to Laguardia ticket cost 280.0
Air Queens 114 10.4.17 13.55 from Kennedy to Laguardia ticket cost 250.0
Air Queens 116 10.4.17 14.39 from Kennedy to Laguardia ticket cost 190.0
```

```
Type the number of the flight you wish to book:
112
Successfully Booked ticket.
Do you want to book or cancel a flight? Answer Yes of No:
Yes
Enter C to cancel, K for a flight from Kennedy, or L for a flight from Laguardia
L
Enter the day in October that you want to fly:
6
Enter an hour you would like to fly (in range 1 - 24)
9
Here are available flights:
Air Queens 101 10.6.17 6.58 from Laguardia to Kennedy ticket cost 140.0
Air Queens 103 10.6.17 7.13 from Laguardia to Kennedy ticket cost 160.0
Air Queens 105 10.6.17 8.57 from Laguardia to Kennedy ticket cost 180.0
Air Queens 107 10.6.17 9.27 from Laguardia to Kennedy ticket cost 210.0
Air Queens 109 10.6.17 10.19 from Laguardia to Kennedy ticket cost 170.0
Air Queens 111 10.6.17 11.55 from Laguardia to Kennedy ticket cost 220.0
Air Queens 113 10.6.17 12.55 from Laguardia to Kennedy ticket cost 230.0
Type the number of the flight you wish to book:
107
Successfully Booked ticket.
Do you want to book or cancel a flight? Answer Yes of No:
Yes
Enter C to cancel, K for a flight from Kennedy, or L for a flight from Laguardia
C
Here are the tickets you have booked:
  1 Donald Duck booked on Air Queens 112 10.4.17 12.08 from Kennedy to Laguardia cost 280.0
  2 Donald Duck booked on Air Queens 107 10.6.17 9.27 from Laguardia to Kennedy cost 210.0
Type the number of the ticket you wish to cancel:
2
Airline Air Queens refunds $210.0 to Donald Duck
Do you want to book or cancel a flight? Answer Yes of No:
No
Thank you for booking with Air Queens
Here is a list of your bookings:
Donald Duck booked on Air Queens 112 10.4.17 12.08 from Kennedy to Laguardia cost 280.0
```