AIRPORT MODEL IN-CLASS ACTIVITY (5 PTS)

FULL NAMES:

NOTES:

- You only need to fill out the attributes and methods of a class one time. After that, you can just show a box with the class name.
- You only need to write the Java code fragments once.
- Do NOT show or write getters/setters.

1. a) Create a UML class diagram that corresponds to the following text.
b) Write the corresponding Java code.

A Passenger has a name, email address, phone number, and a ticket. Each Ticket has an origin, a destination, a gate name, and a seat assignment.

2. a) Create a UML class diagram.

   b) Write the corresponding Java code.

There are two types of planes: Jets and Turboprops. All Planes have an ID, a location (the type is Gps, and you should assume that it has already been written), and status. All Planes can take off and land. Jets have a method to adjust their exhaust output. Turboprops have a method to adjust their propeller speed.

3. a) Create a UML class diagram.
   b) Write the corresponding Java code.

An Airport has a name, an IATA code, and a (GPS) location. It also has one or more Runways, each of which has a name, a location, and zero or one Planes. The Airport has zero or more Terminals. We'll flesh out Terminal in another problem, so just put a box with the class name for now. The Airport also has zero or more Passengers.

4. Create a UML class diagram.

A terminal has a name and location. A terminal is composed of one or more Zones. Each Zone has a name, a location, and a protected method called cleanup(). There are two types of zones: Gates and Stores. A Gate is a special Zone, because it can have zero or one plane (as defined in problem 2 above).

A16
A17
United
Mia Mozarrella
A15
Payphones
Courtesy Phone
A12
A13
A10
Rebar
Dollar Rent A Car
Goodfellows Shoe Shine
A2
A4
Delta Cargo
Chase
US Airways
Vino Volo
A5
Delta
Air Serv Security
A1
A3

5. Create a UML class diagram.

Any Zone in a terminal can request a cleaning by calling the static requestCleanup() method in the MaintenanceUtility class.

6. a) Create a UML class diagram.
   b) Write the corresponding Java code.

{JSON}
JavaScript Object Notation

Every Store in the airport must be capable of charging outrageous prices, and so each Store must implement the OutrageousPriceCharger interface. This interface requires that the getPriceCatalogJson() method is implemented. The Terminal uses the method to ensure that proper gouging is taking place.

7. Let's pull this all together in an overall UML class diagram (no attributes, no methods, no code).

- An airport has 1 or more Runways, 0 or more Terminals, and 0 or more Passengers.
- Each Runway may have a plane.
- A Passenger has a Ticket.
- A Terminal is composed of 1 or more Zones. Terminal depends on the OutrageousPriceCharger interface.
- All Zones can use MaintenanceUtility.
- There are two types of Zones: Gates and Stores.
- Stores must meet the OutrageousPriceCharger interface.
- Gates can have zero or one Planes.
- There are two types of Planes: Jets and Turboprops.