Lab 4 Assignment

Lab Session: Class Modeling

IT314 - Software Engineering

Divyakumar Patel

202201469

Q1

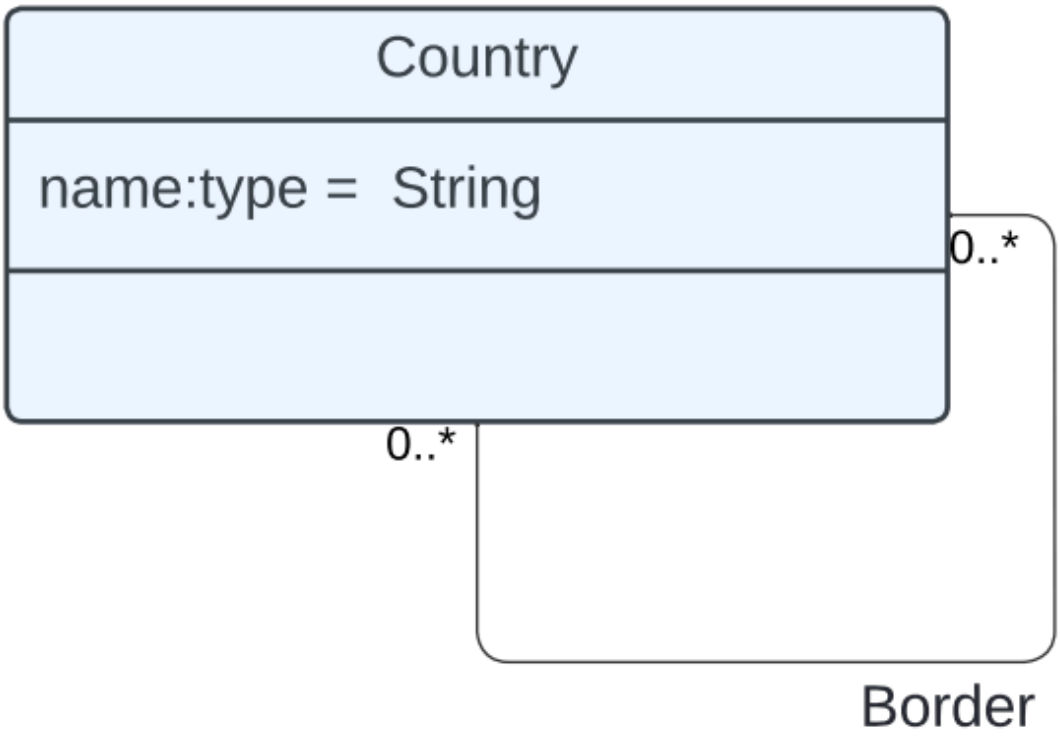
Prepare a class diagram for the following object diagram that shows a portion of Europe.

ANSWER:

Class Diagram Structure:

- **Class Name**: `Country`
- **Attributes**:
  - `name: String`
- **Associations**:
  - `Borders` : A many-to-many association between the `Country` class.

Diagram Representation:



- Each country (Spain, France, Belgium) is an instance of the `Country` class, and the "Borders" relation is an association between instances of this class.
- This class diagram generalizes the structure shown in your object diagram by abstracting individual countries into a more general "Country" class and the "Borders" relationship.

Q2

Prepare a class diagram for object diagram given in Figure -2. Explain your multiplicity decisions. What is the smallest number of points required to construct a polygon? Does it make a difference whether or not point may be shared between polygons? Your answer should address the fact that points are ordered.

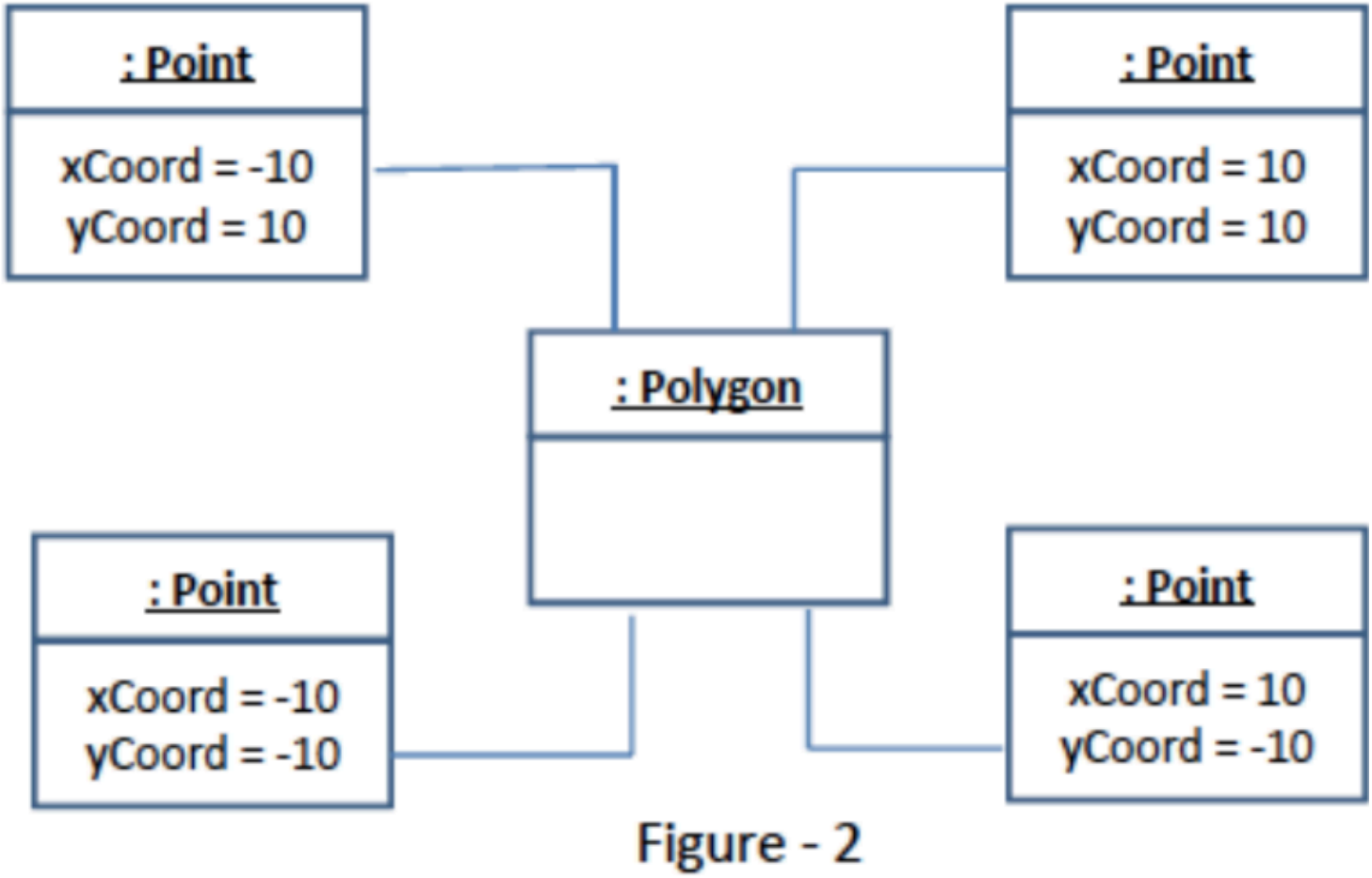


Figure - 2

ANSWER:

Classes

- **Point**: Represents coordinates with `xCoord` and `yCoord` attributes.

- **Polygon**: Defined by multiple `Point` objects, as shown in the diagram.

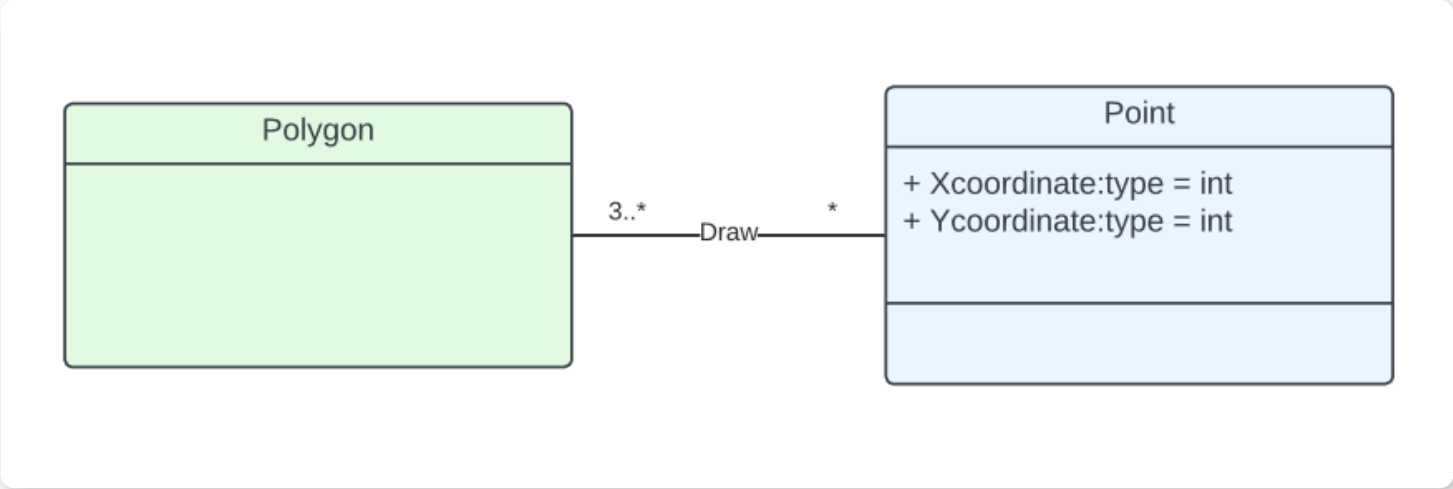
Attributes

- **Point** Class:
  - `xCoord: int`
  - `yCoord: int`
- **Polygon** Class:
  - The `Polygon` class itself does not seem to have additional attributes based on the diagram.

Relationships (Associations)

- **Polygon and Point**:
  - A polygon has a **one-to-many** relationship with points, and the points must be **ordered** to form the shape.
  - **Multiplicity**: A polygon needs at least 3 points.
- **Shared Points**:
  - Points can be shared between polygons, making the relationship **many-to-many** (multiple polygons can share points).

Class Diagram Representation



Explanation of Multiplicity:

- **Polygon to Point**:
  - A polygon has at least 3 points (hence `1..*` cardinality).

    - A point can belong to multiple polygons (many-to-many relationship).
- **Ordered Points**:
    - The relationship between `Polygon` and `Point` is ordered, meaning the order of the points matters when forming the shape.

Smallest Number of Points:

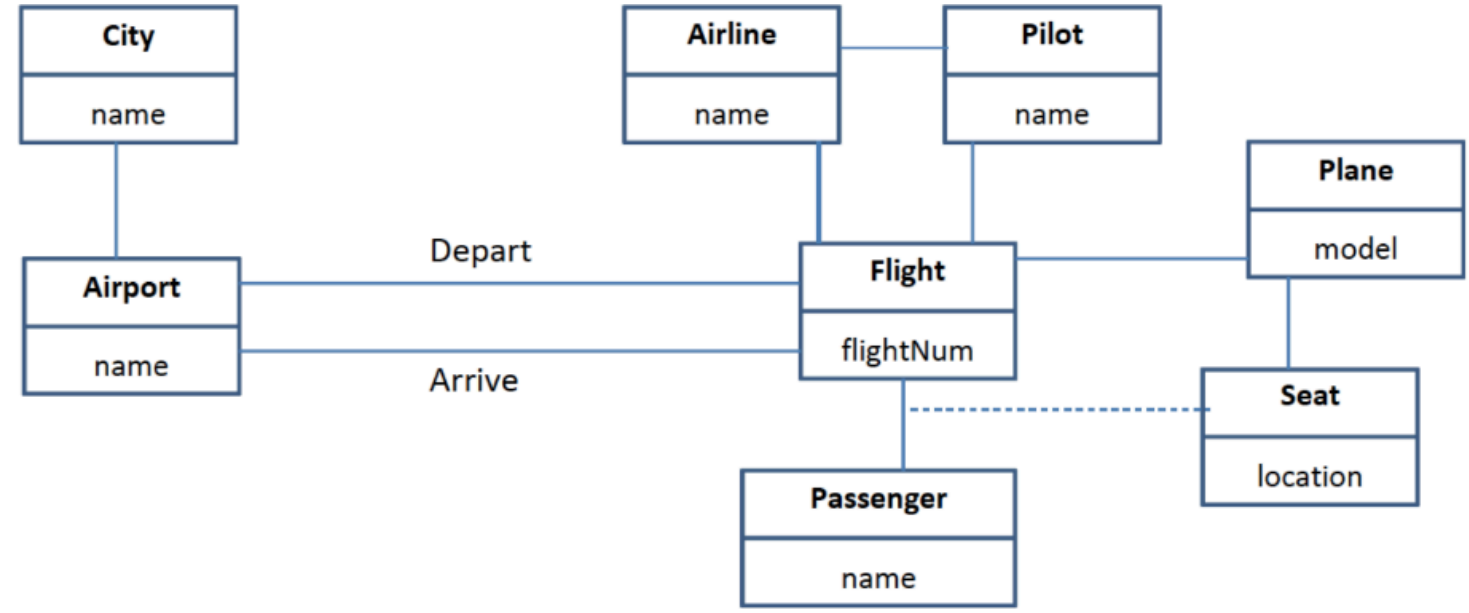
- The smallest number of points required to construct a polygon is **3**, as a polygon needs at least three vertices to form a closed shape (a triangle).

Sharing Points Between Polygons:

- Yes, points can be shared between polygons (e.g., neighboring polygons on a mesh or grid may share vertices). This does not affect the validity of the polygon, but it may impact the specific shape depending on how points are shared and ordered.

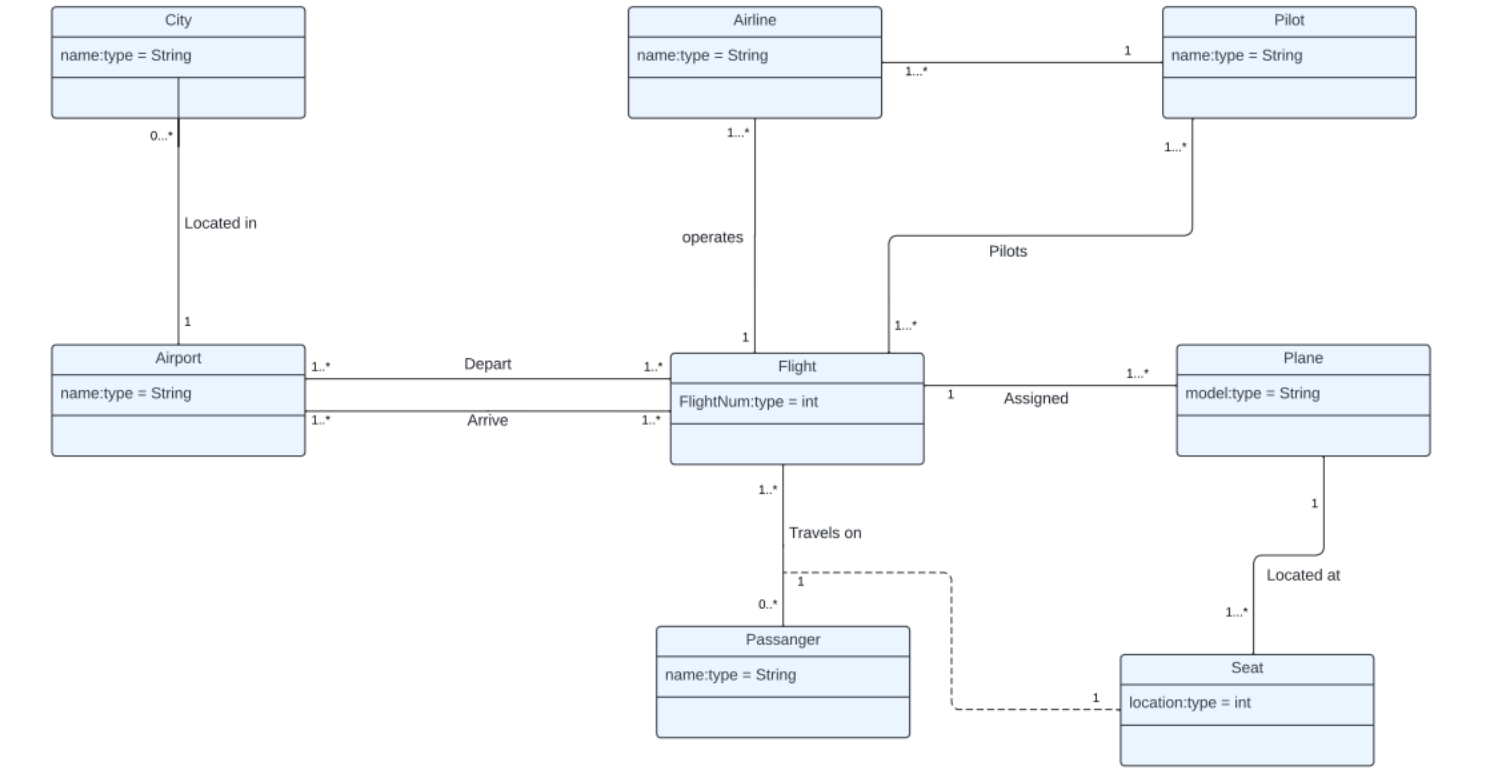
Q3

Figure 3 is a partially completed class diagram of an air transportation system. Add multiplicities in the diagram. Also add association names to unlevelled associations.



ANSWER:

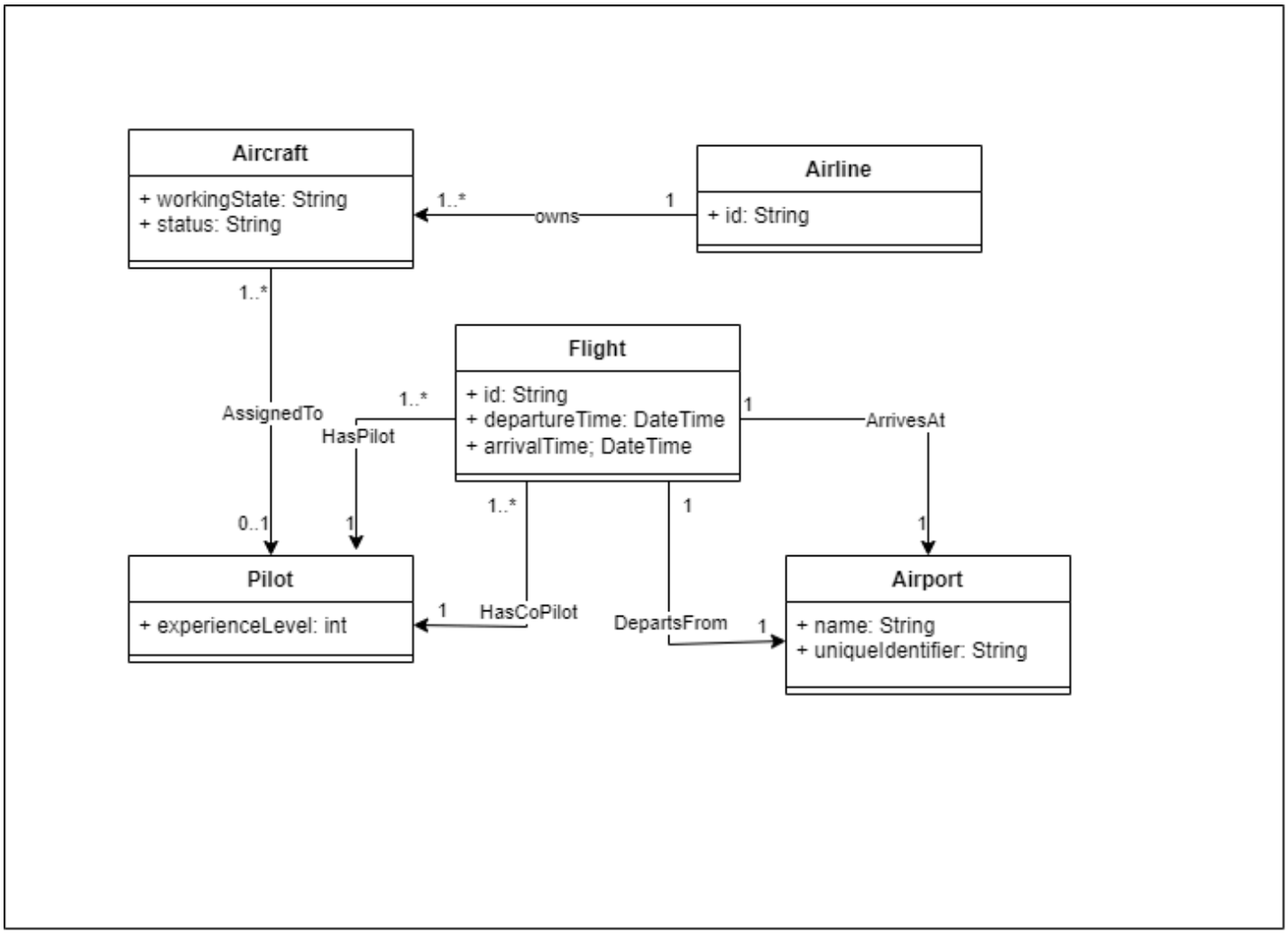
Class Diagram Representation:



Q4

We want to model a system for management of flights and pilots. An airline operates flights. Each airline has an ID. Each flight has an ID a departure airport and an arrival airport: an airport as a unique identifier. Each flight has a pilot and a co-pilot, and it uses an aircraft of a certain type; a flight has also a departure time and an arrival time. An airline owns a set of aircrafts of different types. An aircraft can be in a working state or it can be under repair. In a particular moment an aircraft can be landed or airborne. A company has a set of pilots: each pilot has an experience level: 1 is minimum, 3 is maximum. A type of aeroplane may need a particular number of pilots, with a different role (e.g.: ca0ptain, co-pilot, navigator): there must be at least one captain and one co-pilot, and a captain must have a level 3.

ANSWER:


Aircraft
+ workingState: String
+ status: String
1..*
owns
1
Airline
+ id: String
1..*
AssignedTo
1..*
HasPilot
Flight
+ id: String
+ departureTime: DateTime
+ arrivalTime; DateTime
1
ArrivesAt
0..1
1
1..*
1
1
Pilot
+ experienceLevel: int
1
HasCoPilot
DepartsFrom
1
Airport
+ name: String
+ uniqueIdentifier: String