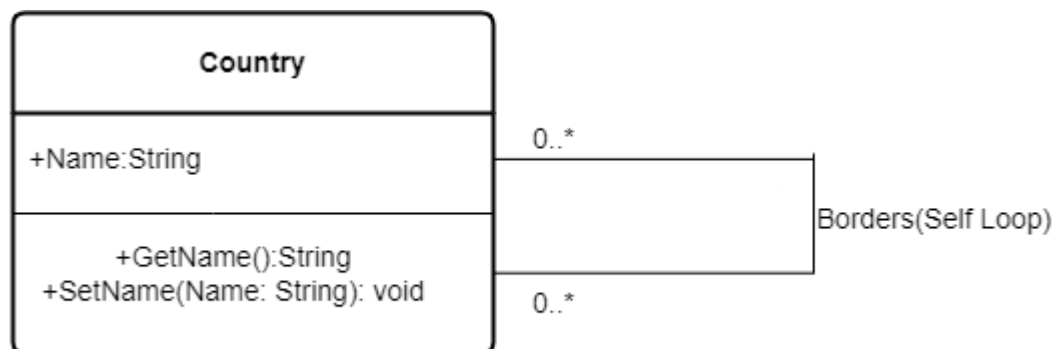# SOFTWARE ENGINEERING
LAB 4 - Class Diagram
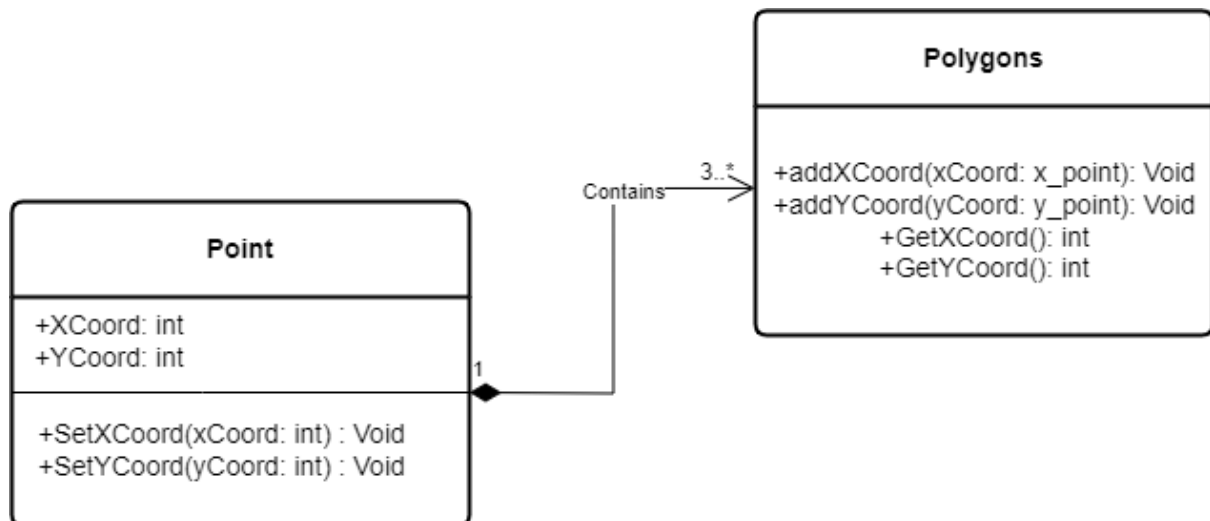
Name : Zeel Boghara
ID : 202201201

Q1 : Prepare a class diagram for the following object diagram that shows a portion of Europe.


```
┌─────────────────────────────────────┐
│              Country                 │
├─────────────────────────────────────┤
│ +Name:String                        │
├─────────────────────────────────────┤
│        +GetName():String            │
│  +SetName(Name: String): void       │
└─────────────────────────────────────┘
```
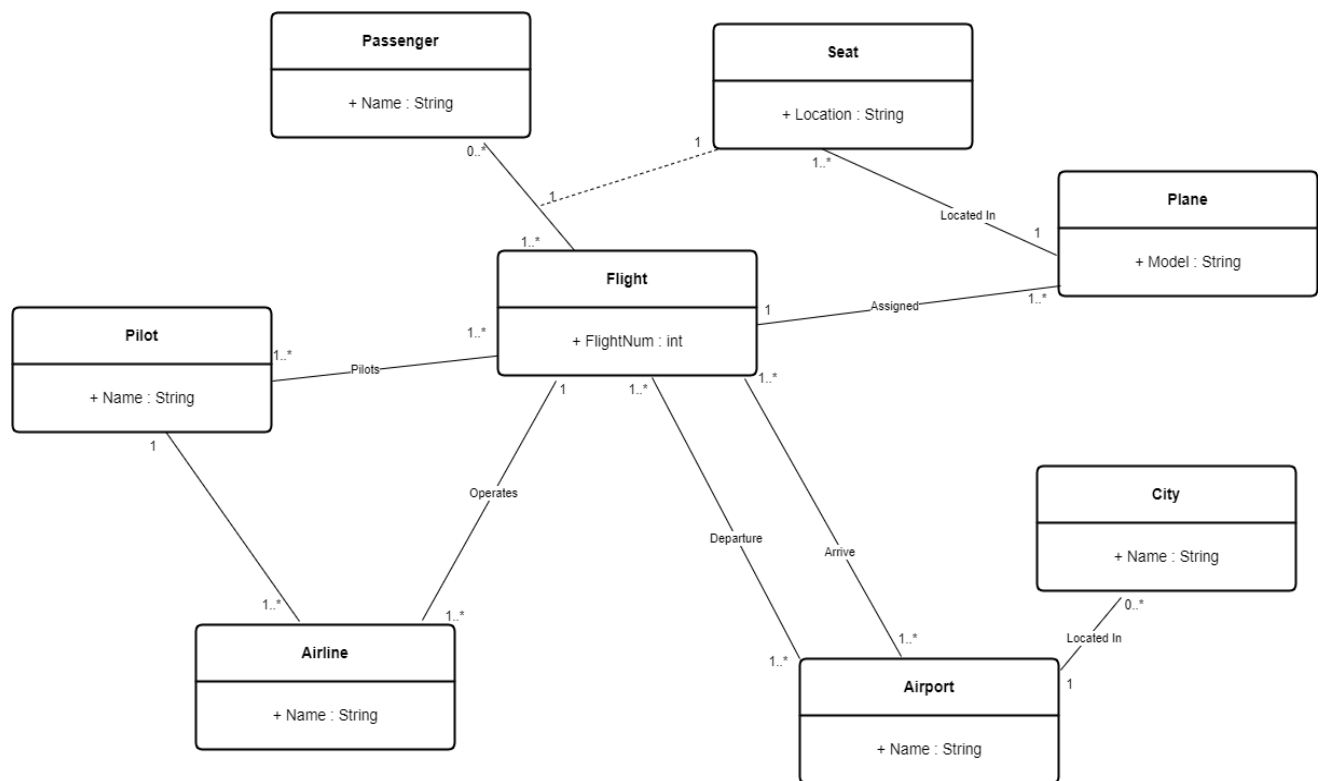
0..*

0..*

Borders(Self Loop)

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.

Point

+XCoord: int
+YCoord: int

+SetXCoord(xCoord: int) : Void
+SetYCoord(yCoord: int) : Void

1

Contains

3..*

Polygons

+addXCoord(xCoord: x_point): Void
+addYCoord(yCoord: y_point): Void
+GetXCoord(): int
+GetYCoord(): int

❖ 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.

UML Class Diagram

Passenger
+ Name : String

Seat
+ Location : String

Plane
+ Model : String

Flight
+ FlightNum : int

Pilot
+ Name : String

City
+ Name : String

Airline
+ Name : String

Airport
+ Name : String

Relationships and multiplicities:
- Passenger 0..* —— 1..* Flight
- Seat 1 ---- 1..* (dashed to Flight)
- Seat 1..* — Located In — 1 Plane
- Flight 1 — Assigned — 1..* Plane
- Pilot 1..* — Pilots — 1..* Flight
- Pilot 1 — Operates — 1..* Airline
- Flight 1..* — Operates — 1..* Airline
- Flight 1 — Departure — 1..* Airport
- Flight 1..* — Arrive — 1..* Airport
- City 0..* — Located In — 1 Airport

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.:

captain, co-pilot, navigator): there must be at least one captain and one co-pilot, and a captain must have a level 3.

Airline
+ Id : String

Airline 1 —— owns —— 1..* Aircraft

Aircraft
+ WorkingState: String + Status : String

Flight
+ id: String + departureTime : Date/Time + arrivalTime : Date/Time

Airport
+ Name : String + uniqueIdentifier : String

Pilot
+ experienceLevel: int

Flight 1 —— ArrivesAt —— 1 Airport

Flight 1 —— DepartsFrom —— 1 Airport

Flight 1..* —— HasPilot —— 1 Pilot

Flight 1..* —— HasCoPilot —— 1 Pilot

Aircraft 1..* —— AssignedTo —— 0..1 Pilot