OBSERVER

EVER HINOJOSA AGUIRRE

OBSERVER

- FOR THE OBERSERV PATTERN I TRIED TO MIMIC THE NOTIFICATION SIGNALS OF AIRPORTS IN MONTERREY, SEEING AS THE CONTROL TOWERS CAN BENEFIT FROM BEING INFORMED WHEN A FLIGHT IS ARRIVING OR LEAVING I THINK THE COMMUNICATION STYLE OF THE PATTERN CAN FIT A SOFTWARE OF THIS TYPE. HAVING A SUPERVISOR AS A SUBJECT THAT NOTIFIES THE AIRPORT OBSERVERS ABOUT FLIGHTS AND THEIR INFORMATION.

Airport.java | ControlTower.java | Observer.java | Subject.java | Supervisor.java

```java
import java.util.ArrayList;

public class Supervisor implements Subject{
    private List<Observer> observers = new ArrayList<>();
    private String flightInfo;

    public void setFlightInfo(String flightInfo) {
        this.flightInfo = flightInfo;
        notifyObservers();
    }

    @Override
    public void attach(Observer observer) {
        observers.add(observer);
    }

    @Override
    public void detach(Observer observer) {
        observers.remove(observer);
    }

    @Override
    public void notifyObservers() {
        for (Observer observer : observers) {
            observer.update(flightInfo);
        }
    }
```

Airport.java | ControlTower.java | Observer.java | Subject.java | Supervisor.java

```java
import java.util.Objects;

public class Airport implements Observer{
    private String airport;

    public Airport(String name) {
        this.airport = name;
    }

    @Override
    public void update(String flightInfo) {
        System.out.println("ControlTower " + airport + " received flight info: " + flightInfo);
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Airport other = (Airport) obj;
        return Objects.equals(airport, other.airport);
    }
}
```