AIRLINE TICKET MANAGEMENT

(DONE BY: A.SUDARSHAN (2019B4A70744P) and RAJAN SAHU (2019B4A70572P))

ANALYSIS OF PRICIPLES OF PROGRAMMING

1. **Encapsulate what varies:**
    a. In the Airline.java we have separated those members that will vary among the airline companies like name, code, set of flights and staffs into Airline classes and created child classes from Airline class for each Airline company.

2. **Favour composition over inheritance:**
    a. In Airline.java, we have favoured composition by adding list of Flights rather than inheriting flight class from airline class. Similar thing has been done for staff objects as well (Reference of staff objects in airline)

3. **Strive for loose coupling between objects that interact:**
    a. We have tried to make the objects as independent as possible, but still there are places like Booking and cancellation where objects of Flights, Booking and Seat might have strong coupling
    b. Implementing loose coupling would have helped us in adding features or changes, if necessary, in a more efficient way (handling only those class objects)

4. **Program to interface and not implementation:**
    a. We could have encapsulated what varies in Airline class by making Airline as interface instead of class. However, since we wanted few members that required to be accessed by other classes such as booking, cancellation, etc. we used Airline as a class rather than an interface

DESIGN PATTERN ANALYSIS

Behavioural pattern:

Behavioural pattern is about identifying common communication patterns between objects

Eg. Iterator, State, Strategy, Observer, Command, Chain of Responsibility

Here in our project, there is common communication of various airline companies (as varieties of airline companies). The name, the code, set of flights and set of staffs vary. So, we can separate these attributes that vary including methods like update flight status,etc and put them in an interface, and have different implementations for these functions. This is the strategy pattern. We haven't implemented this, or any design pattern that we know of.