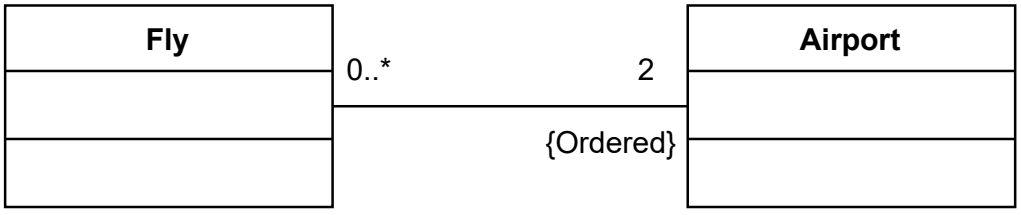
Exercise 1

Design the class diagram for a flight from the departure airport to the arrival airport.

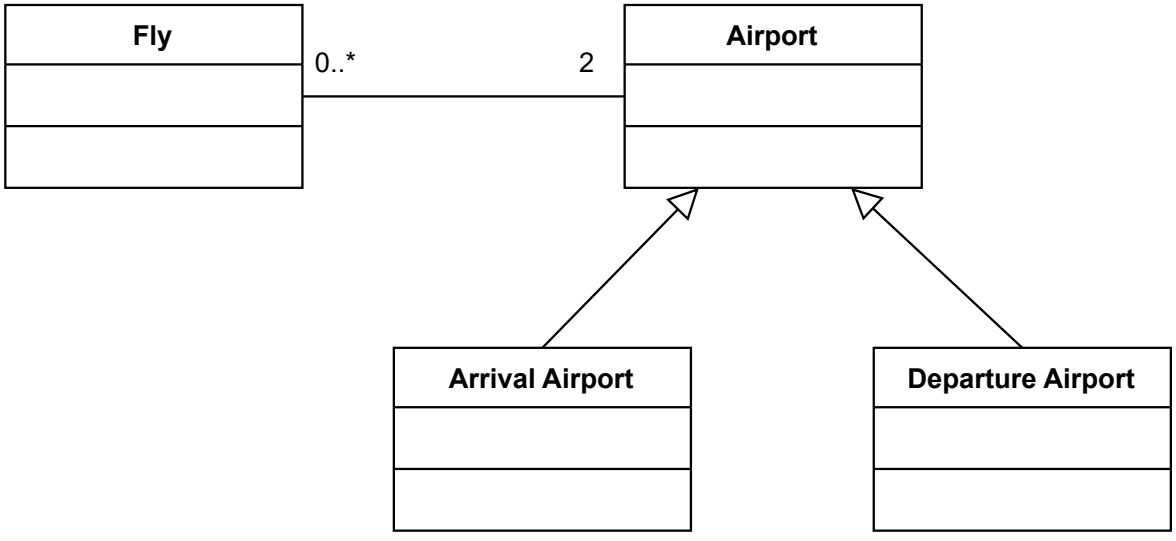
Solution$_1$



Ordered

{Ordered} is a constraint applied to a collection of instances where the order of elements matters. For example, in this case, we have a collection of airports where the first element represents the departure airport, and the second represents the arrival airport.

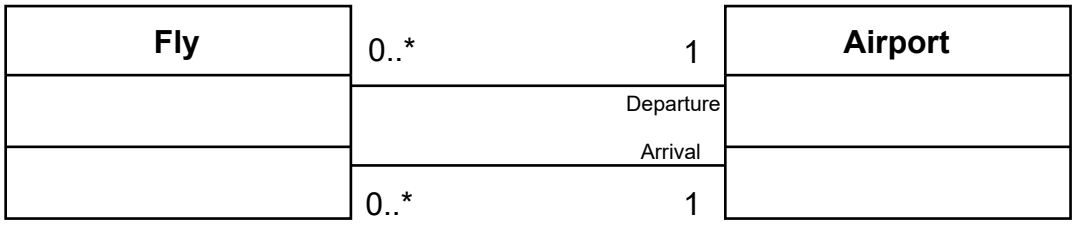
Solution$_2$



Note

The issue with using inheritance is that it duplicates instances, as each airport would require a separate instance for departure and another for arrival.

Solution$_3$



Role

A role defines the specific function a class plays in a relationship , a class might have more than just one role , in our case an airport plays 2 roles : departure or arrival airport.

Exercise 2

Give the implementation for an application that creates only one instance of the pilot with card code 100 using singleton.

Solution

Java Code:

```java
public class Pilot {

    private static volatile Pilot instance = null;
    private int card;

    private Pilot(int card) {
       this.card = card;
       System.out.println("Pilot Instance");
    }

    public static Pilot getInstance() {

        Pilot result = instance;

        if (result == null) {

            synchronized(Pilot.class) {
            result = instance;

            if (result == null) {
                instance = result = new Pilot(100);
            }

            }
        }

        return result;
    }


}
```

```java
public class Main {

     public static void main(String[] args) {

        Runnable task = () -> {
         System.out.println("Running on thread: " + Thread.currentThread().getName());
       Pilot p1 =  Pilot.getInstance();
        };

        Thread thread1 = new Thread(task);
      Thread thread2 = new Thread(task);
      Thread thread3 = new Thread(task);

      thread1.start();
      thread2.start();
      thread3.start();
     }
}
```

Output:

```
Running on thread: Thread-0
Running on thread: Thread-1
Pilot Instance
Running on thread: Thread-2
```