Reto 2

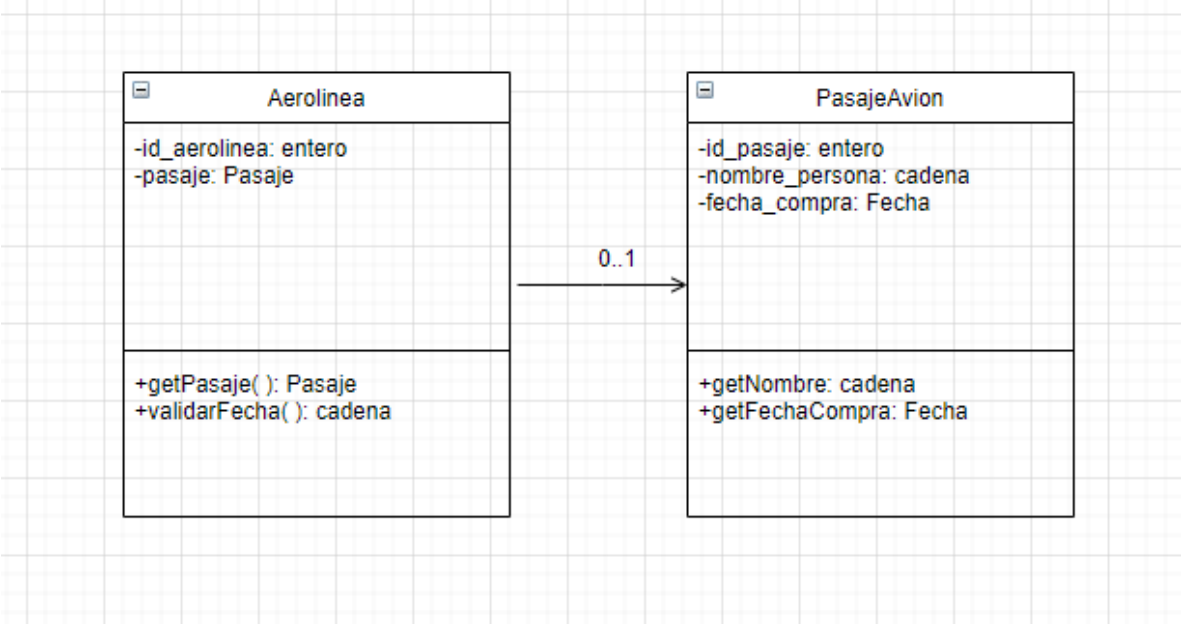
Objetivo

El objetivo de este reto es que el estudiante reconozca y aplique los elementos básicos del paradigma de la programación orientada a objetos en un escenario abstraído de la cotidianidad.

Contexto

La aerolínea “Viv Col” en los últimos meses ha presentado problemas para validar la fecha de vencimiento de los pasajes de sus usuarios de manera online. Por ende, necesita un programa que complemente su sistema, el cual verifique que tenga máximo 4 meses de antigüedad la fecha de compra y de acuerdo a esto, le notifique al usuario si su pasaje es válido.

Considere el siguiente diagrama de clases para la implementación de la clase **Aerolínea** y **PasajeAvion.java**



NOTA: Las clase deben llamarse Aerolinea.java y PasajeAvion.java

Reto

Implemente una función llamada validarFecha(), la cual dada una instancia de PasajeAvion, verificará la fecha de compra del pasaje y dependiendo del valor de este, se pueden presentar los siguientes casos:

a) Si la fecha de compra tiene una antigüedad mayor a 4 meses, se le debe mostrar al usuario el siguiente mensaje: “Señor usuario, su pasaje esta vencido y no puede ser usado en esta ocasión”.
b) Si la fecha de compra tiene una antigüedad menor o igual a 4 meses, pero mayor o igual a 2 meses, se le debe mostrar al usuario el siguiente mensaje: “Señor usuario, su pasaje es válido, pero está próximo a vencer. Debe ser usado antes de que caduque”.
c) Si la fecha de compra tiene una antigüedad menor a 2, se le debe mostrar al usuario el siguiente mensaje: “Señor usuario, su pasaje es válido y aún tiene más de dos meses para darle uso”.

Adicionalmente, use la siguiente imagen como referencia para la construcción de la clase Aerolinea y Pasaje con sus atributos y métodos necesarios.

```java
public class Aerolinea {
    private int id_aerolinea;
    private PasajeAvion pasaje;

    public Aerolinea(int id_aerolinea, PasajeAvion pasaje){


    }

    public static void main(String[] args) {


    }

    int getId_aerolinea() {
        return id_aerolinea;
    }

    void setId_aerolinea(int id_aerolinea) {
        this.id_aerolinea = id_aerolinea;
    }

    PasajeAvion getPasaje() {
        return pasaje;
    }

    void setPasaje(PasajeAvion pasaje) {
        this.pasaje = pasaje;
    }

}
```

```java
import java.sql.Date;

public class PasajeAvion {
    private int id_pasaje;
    private String nombre_persona;
    private Date fecha_compra;

    public PasajeAvion(int id_pasaje, String nombre_persona, Date fecha_compra) {


    }

    int getId_pasaje() {
        return id_pasaje;
    }
    void setId_pasaje(int id_pasaje) {
        this.id_pasaje = id_pasaje;
    }
    String getNombre_persona() {
        return nombre_persona;
    }
    void setNombre_persona(String nombre_persona) {
        this.nombre_persona = nombre_persona;
    }
    Date getFecha_compra() {
        return fecha_compra;
    }
    void setFecha_compra(Date fecha_compra) {
        this.fecha_compra = fecha_compra;
    }

}
```

Casos de prueba

Finalmente, para verificar el funcionamiento del programa se sugiere considerar los siguientes casos de prueba:

# CASO DE PRUEBA	DATO DE ENTRADA	SALIDA ESPERADA
1	**PasajeAvion** Id_pasaje: 0 nombre_persona: “Juan Manuel Zuluaga” fecha_compra: 3/12/2021 **Para el ejemplo se toma como fecha actual: 8/12/2021 (MM/dd/AAAA)**	Señor usuario, su pasaje esta vencido y no puede ser usado en esta ocasión
2	**PasajeAvion** Id_pasaje: 1 nombre_persona: “Lida Patricia Henao” fecha_compra: 6/12/2021 **Para el ejemplo se toma como fecha actual: 8/12/2021 (MM/dd/AAAA)**	Señor usuario, su pasaje es válido, pero está próximo a vencer. Debe ser usado antes de que caduque

<table>
<tr>
<td>3</td>
<td>
<table>
<tr><th>PasajeAvion</th></tr>
<tr><td>Id_pasaje: 2
nombre_persona:
“Neider Gaviria”
fecha_compra: 7/17/2021</td></tr>
</table>

Para el ejemplo se toma como fecha actual: 8/12/2021 (MM/dd/AAAA)
</td>
<td>Señor usuario, su pasaje es válido y aún tiene más de dos meses para darle uso</td>
</tr>
</table>

Entrega:

1. Suba a la plataforma un archivo con el nombre de **Aerolínea.java** y **PasajeAvion.java**, este nombre debe de respetarse, dado que, si no se nombre de dicha manera no se tendrá en cuenta para la calificación del reto.

2. **Importante:** Los métodos deben de llamarse **exactamente igual** a como se muestra en el ejemplo de la estructura del código.

3. **Importante:** Las salidas deben ser tal cual se muestran en los casos de pruebas. De lo contrario, el sistema no lo reconocerá.