RECUPERATORIO INTEGRAL IPOO 2018

Se desea implementar un sistema para una aerolínea (nombre de la aerolineas, colección de vuelos, colección de pasajeros) que permita gestionar información de sus vuelos, aviones, pasajeros y destinos. Los diferentes destinos se programan con anticipación para ser publicitados durante el transcurso del año, la información que se almacena del destino es el nombre del lugar y una descripción que sera utilizada por los operadores para informar a sus pasajeros.
De los pasajeros se almacena el número de pasaporte, el número de documento, nacionalidad, el nombre y el apellido.
De los aviones se almacena el número de avión, el número de plazas para cada una de las clases ejecutiva y económica.
De los vuelos se desea almacenar el número de vuelo, cantidad de plazas disponibles en cada clase, hora de partida, hora de llegada al destino, el destino, el avión asignado, el importe y la colección de pasajeros que van a realizar ese vuelo. Los vuelos pueden ser Nacionales o Internacionales. De los vuelos internacionales se almacena la **cantidad** (solo la cantidad) de escalas que deben realizarse antes de llegar a destino. El costo de un pasaje (importe final del vuelo) se calcula sobre el importe del vuelo y si el pasajero es Argentino se aplica el 21% de IVA caso contrario el impuesto no es aplicable. Por otro lado si el vuelo es internacional se debe incrementar el costo con los gastos de utilización de rutas internacionales que se calcula de la siguiente manera:

costo vuelo / cantidad de escalas * 0,7 .

Ejercicio

1. Crear las siguientes clases: Aerolinea, Avión, Pasajero, Destino, Vuelo y la jerarquía correspondiente. Para cada una de las clases: definir las variables instancias, Implementar el método constructor, __toString y los métodos de acceso de cada una de las variables instancias definidas en la clase.
2. En la clase Vuelo implementar el método ***calcularImporte*** y redefinir según sea necesario.
3. En la clase Aerolinea implementar el método **configurarVuelo** que recibe por parámetro un objeto Destino, un objeto avión y un arreglo asociativo con las siguientes claves (partida, hora de llegada al destino, importe). Este método es el encargado de crear un objeto Vuelo y dejarlo listo para ser comercializado. La cantidad de plazas disponibles del vuelo va a coincidir con la cantidad de plazas del avión y la colección de pasajeros que van a realizar el vuelo debe crearse vacía, El método retorna la instancia de vuelo creada. (Ayuda: puede utilizar como numero de vuelo el subíndice de la colección de vuelos de la aerolinea)
4. Implementar el método ***venderVuelo*** que recibe por parámetro un número de vuelo, el objeto pasajero, la clase en la que desea viajar (ejecutiva/económica) y retorna el importe que debe abonar el pasajero si la la operación pudo realizarse con éxito. Este método debe chequear las plazas disponibles, actualizarlas si la venta se realiza y vincular el pasajero a la colección de pasajeros del vuelo,
5. Implementar un script TestAerolinea donde:
    1. Cree 2 instancias Destinos: d1, d2.
    2. Cree 2 instancias de Avion: av1 ,av2. La cantidad de plazas de av1 es 1
    3. Cree 1 instancia de Aerolinea-
    4. Cree 2 Instancias de Pasajero: p1, p2
    5. Invoque al método **configurarVuelo (d1,av1)** , **configurarVuelo (d2,av2)** visualizar las respuestas a la invocación de los métodos.
    6. Sea vuelo1 la instancia retornada por la invocación al método **configurarVuelo (d1,av1) donde** numero de vuelo es 001**.** Invocar al método ***venderVuelo*** con los siguientes parámetros***:*** ***venderVuelo(*** 001, p1,'Economica)' y visualizar el resultado.
    7. Sea vuelo1 la instancia retornada por la invocación al método **configurarVuelo (d1,av1).** Invocar al método ***venderVuelo*** con los siguientes parámetros***:*** ***venderVuelo(*** vn001, p2,'Economica)' y visualizar el resultado.