Programación I

Examen Convocatoria Extraordinaria 4-Febrero-2015

La aerolínea Oceanic Airlines nos ha pedido que realicemos algunas tareas de gestión de sus vuelos. Para ello, en el programa principal (main) tenemos las siguientes estructuras de datos que almacena la siguiente información:

- *vAeropuertos*, es un array de Strings que almacena la lista de aeropuertos en los que opera la compañía.
- *vPasajeros*, almacena un lista de objetos Pasajero, cuya definición es (con sus correspondientes *getters y setters*):
    - DNI, que identifica al pasajero
    - Nombre
    - Edad
- *vDistanciaMillas*, almacena la distancia en millas entre dos aeropuertos. Está representado por un array bidimensional. Por claridad, las tenéis en la siguiente tabla:

	LAX	SYD	BIO	SIN
LAX	0	7503	5737	8772
SYD	7503	0	2753	9023
BIO	5737	2753	0	7005
SIN	8772	9023	7005	0

La clase Pasajero **la tenéis implementada** al final del examen.

Para llevar a cabo nuestro encargo, se debe implementar la clase Billete.

[1,5 puntos] Clase Billete

La clase Billete representa un billete vendido de la aerolínea. La información que se registra en cada billete, en relación al pasajero es:

- DNI: una cadena de caracteres que identifica al pasajero.
- Número de asiento: cadena de caracteres compuesta por una letra y un número de máximo 2 cifras (“A13”).
- Precio del billete: número real, entre 20 y 1200 euros.
- Código de vuelo, que se forma de la siguiente forma:
    - XXX, tres caracteres que representan aeropuerto de origen del vuelo.
    - 999, tres dígitos que representan el vuelo.

    - XXX, tres caracteres que representan el aeropuerto del destino del vuelo.

- Puntos, es un número real positivo que indica el número de puntos obtenidos por el pasajero en el vuelo, en función de las millas recorridas.

Para manejar estos datos, la clase Billete facilita los siguientes métodos:

- **Constructor** sin argumentos, que inicializa todos los atributos con valores válidos por defecto (cada programador debe elegir cuáles son estos valores)

- **Constructor** con parámetros, que inicializa todos los atributos con valores pasados como parámetro, comprobando que son valores válidos. En caso de que no lo sean, se inicializarán con valores por defecto (iguales a los del constructor anterior).

- Métodos **set y get** para todos los atributos de la clase, excepto el DNI, que no se puede modificar. IMPORTANTE: cada vez que se cambie el valor de un atributo de la clase, se debe comprobar que el nuevo valor a asignar sea válido, en caso contrario se dejará el valor que tenía antes (no se debe mostrar ningún mensaje por pantalla)

- Además se deberán programar los siguientes métodos get, obtenidos a partir del código del empleado (XX99X, *pe."SYD815LAX"*):

    - **getOrigen**, devuelve la cadena con los dos caracteres que conforman el código de la oficina (en el ejemplo, devolvería "SYD")

    - **getNumVuelo**, devuelve el entero que representa el código de empleado (en el ejemplo, devolvería 815)

    - **getDestino**, devuelve el aeropuerto de destino (en el ejemplo devolvería 'LAX')

- Método **leer**, que lee por teclado todos los atributos de la clase (salvo los puntos, que los pone a cero), y comprueba que sean correctos. En caso de no serlo los seguirá pidiendo una y otra vez. También debe de calcular los puntos en función de las millas de distancia del viaje. Este método recibe como parámetros los arrays necesarios para realizar los cálculos.

[1 puntos] Método asignarPuntosAlBillete

En base a las millas de distancia del viaje, se le asigna un número de puntos al billete a través de la siguiente fórmula:

puntos = número de millas * 0,75

Las distancias en millas correspondientes a cada uno de los aeropuertos están almacenadas en el array *vDistanciaEnMillas*. Primero, tenéis que desarrollar un método que se llame *buscarPosicionAeropuerto*. Este método buscará la posición dentro del array *vAeropuertos*. Esta posición sirve como índice del array *vDistanciaEnMillas*. Es decir, el aeropuerto "LAX" está en la posición 0 del array *vAeropuertos*, y en la posición 0 de las dos coordenadas del array *vDistanciaEnMillas.* Todos estos arrays lo teneís que recibir como parámetros de la función.

Una vez obtenidas las millas, se calcula el número de puntos que se almacenará en el atributo correspondiente.

Clase Gestion

Una vez creada la clase Billete, se deben programar los siguientes métodos para gestionar los pasajeros de los vuelos de la compañía.

[1,5 puntos] Método cargarBilletes

Este método lee por teclado los datos de todas los billetes, que serán un máximo de 1000, y los almacena en el array *vBilletes* en el orden inverso al que se introducen. En cada billete comprobará si está en el array vPasajeros y en caso de que no lo esté pedirá un nuevo DNI. Después de introducir cada billete se preguntará al usuario si quiere introducir más (contemplar las mayúsculas y minúsculas). El proceso finaliza al introducir 1000 billetes o cuando el usuario indique que no quiere introducir más.

Para facilitar el cálculo, crea un método denominado **obtenerPasajeroPorDNI** que reciba un DNI y devuelve el objeto Pasajero dentro del array *vPasajeros* cuyo DNI coincida. En caso de que no existe, debe de devolver el valor *null*.

[2 puntos] Método probabilidadSupervivenciaAvión

Para calcular la prima, una compañía de seguros evalúa las posibilidades de supervivencia que tiene el pasaje (el conjunto de pasajeros). Para ello, calcula la probabilidad en función de edad:

- Si el viajero es menor de 18 años, tiene un 45% de posibilidades de sobrevivir.
- Entre 18 y menos de 25, tiene un 40% de posibilidades de sobrevivir,
- Entre 25 y 35, tienen un 35%
- Entre más de 35 años , un 30% de posibilidades

La probabilidad de supervivencia del avión en caso de accidente, que es la que se necesita para calcular la prima, es la media de las probabilidades todos los pasajeros que viajan en el avión.

Este método recibe como parámetro el códgo de vuelo cuya prima se quiere calcular. Posteriormente, hay que recuperar todos los DNI del array de *vBilletes* cuyo código de vuelo coincida con el número de vuelo que se busca. Finalmente, buscamos el pasajero en el array *vPasajeros* y obtenemos su edad. Calculamos la probabilidad en base a la edad del pasajero y finalmente calculamos la media de las probabilidades de todos los pasajeros.

[2 puntos] Método ordenarPasajeros

Este método debe, dado un número de vuelo, mostrar por pantalla el nombre de los pasajeros ordenados por el número de asiento. Se debe obtener todos los billetes que se han comprado para el vuelo. Luego hay que ordenar esos billetes y finalmente, recuperamos el nombre del pasajero del array *vPasajero*.

[2 puntos] Método estadísticasVuelos

También es importante saber las estadísticas de cada uno de los vuelos para los que se han vendido los billetes. El método debe mostrar, por cada vuelo para el que se han vendido billetes, el número de pasajeros de dicho vuelo y la edad media del pasaje.

Ejemplo de ejecución

```
Introduzca el DNI:
01
Introduzca el DNI:
0001
Introduzca el número de asiento: (X99)
A1
Introduzca el código de vuelo
SYD815LAX
¿Desea seguir introduciendo billetes? (S/N)
S
Introduzca el DNI:
0002
Introduzca el número de asiento: (X99)
A2
Introduzca el código de vuelo
SYD815LAX
¿Desea seguir introduciendo billetes? (S/N)
N
Probabilidad de supervivencia del vuelo SYD815LAX: 32.49999999999999%


Pasajeros del vuelo SYD815LAX
=========================================
A1:    Jack Shephard
A2:    Kate Austen


Estadísticas
=========================================
Vuelo           Pasajeros            Edad media
-----           ---------            ----------
SYD815LAX       2                    36
```

Clase Pasajero

```
/**
 * Clase que representa un pasajero del vuelo.
 */
public class Pasajero {

      private String dni= "";
      private String nombre = "";
      private int edad = 0;
      private double probabilidadSupervivencia = 0;


      /**
       * Constructor vacío.
       */
      public Pasajero() {   }
```

```
	/**
	 * Constructor con parámetros.
	 *
	 * @param dni
	 * @param nombre
	 * @param edad
	 * @param probabilidadSupervivencia
	 */
	public Pasajero(String dni, String nombre, int edad) {
		super();
		this.dni = dni;
		this.nombre = nombre;
		this.setEdad(edad);
	}

	public String getDNI() {
		return dni;
	}

	public void setDNI(String dni) {
		this.dni = dni;
	}

	public String getNombre() {
		return nombre;
	}

	public void setNombre(String nombre) {
		this.nombre = nombre;
	}


	public int getEdad() {
		return edad;
	}


	public void setEdad(int edad) {
		// Si es mayor que 0
		if (edad > 0){
			// Acutalizamos la edad
			this.edad = edad;
		}
	}


	public double getProbabilidadSupervivencia() {
		return probabilidadSupervivencia;
	}
}
```

Clase Gestion

```
public class Gestion{

     public static void main(String[] args){

          // Array que almacena los pasajeros de Oceanic Airlines
          Pasajero[] vPasajeros = {
                    new Pasajero("0001", "Jack Shephard", 38),
                    new Pasajero("0002", "Kate Austen", 25),
                    new Pasajero("0003", "James Ford", 35),
                    new Pasajero("0004", "John Locke", 52),
                    new Pasajero("0005", "Sayid Jarrah", 36),
                    new Pasajero("0006", "Hugo Reyes", 31),
                    new Pasajero("0007", "Claire Littleton", 23),
                    new Pasajero("0008", "Desmond Hume", 37),
                    new Pasajero("0009", "Benjamin Linus", 50),
          };

          // Atributo que guarda la lista de aeropuertos
          // en los que opera la compañía
          String[] vAeropuertos = {
               "LAX", "SYD", "BIO", "SIN"
          };

          // Atributo que almacena la distancia en millas entre
          // los aeropuertos.
          int[][] vDistanciaMillas = {
               {0, 7503, 5737, 8772},
               {7503, 0, 2753, 9023},
               {5737, 2753, 0, 7005},
               {8772, 9023, 7005, 0},
          };

          // Array que almacena los billetes
          Billete[] vBilletes = new Billete[1000];
          cargarBilletes(vBilletes, vPasajeros,
                              vDistanciaMillas, vAeropuertos);
          probabilidadSupervivencia(vBilletes,
                              vPasajeros, "SYD815LAX");
          ordenarPasajeros(vBilletes, vPasajeros, "SYD815LAX");
          calcularEstadisticas(vBilletes, vPasajeros);
     }
}
```