IES HENRI MATISSE
C/ Enric Valor s/n - 46980 Paterna
Telf.961206280- Fax.961206281
46022622@gva.es
www.ieshenrimatisse.es

1r DAM **Ex. Final – 2020-2021 (07/06/2021)** **Módul: PRG**

Ejercicio 1 Cuando los alumnos de segundo presentan su proyecto de fin de curso, la nota del proyecto se calcula de la siguiente manera:

- Cada miembro del tribunal pone su nota (un valor entero entre 0 y 10)
- La nota del proyecto es la media de las notas, sin tener en cuenta la nota la nota más alta ni la nota más baja. Se redondea a dos decimales.

En el siguiente ejemplo, en cada fila aparecen sombreadas las 2 notas que no se tendrían en cuenta para calcular la media.

	Notas del alumno								Nota del proyecto
Alumnos	9	1	3	4	5	6	7	4	4.83
	5	7	6	3	5	8	1	6	5.33
	5	5	5	5	5	5	5	5	5.0
	8	9	4	3	5	8	7	3	5.83
	9	10	8	7	5	8	8	10	8.33
	7	2	6	9	6	5	7	6	6.17

Disponemos de una matriz de notas con una fila por cada alumno/a y una columna por cada miembro del tribunal. Se asegura que cada alumno tendrá al ménos 3 notas, es decir, que la matriz tendrá, al menos, 3 columnas.

Se pide: completar el método calcularNotasFinales para que devuelva un array con las notas de proyecto de cada uno de los alumnos (un elemento por alumno)

Test: En el main se prueba el método con dos matrices. El resultado esperado es el siguiente:

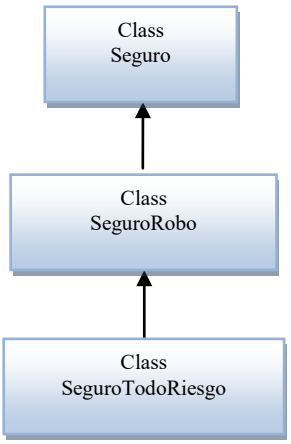
```
[4.83, 5.33, 5.0, 5.83, 8.33, 6.17]
[5.25, 6.25]
```

IES HENRI MATISSE
C/ Enric Valor s/n - 46980 Paterna
Telf.961206280- Fax.961206281
46022622@gva.es
www.ieshenrimatisse.es

Ejercicio 2 Una compañía de seguros para vehículos ofrece a sus clientes tres tipos de seguros:

Seguros (digamos, normales), seguros de robo y seguros a todo riesgo.

Implementar la siguiente jerarquía de clases:



Seguro (ya tiene los atributos, constructor y getter)

Permite almacenar información sobre un seguro básico. Para ello la clase dispone de los siguientes atributos privados:

- anyo (int): Año de vigencia del seguro
- matricula (String): Matrícula del vehículo asegurado.
- incendio (boolean): Indica si el seguro incluye o no cobertura contra incendio.
- lunas (boolean): Indica si el seguro incluye o no cobertura de rotura de lunas
- conductor (Conductor): Es un objeto de la clase conductor. La clase conductor se encuentra implementada en el paquete.

Añade los siguientes métodos (sólo los que se piden):

- **calcularPrecio()**: Devuelve el precio del seguro teniendo en cuenta las siguientes condiciones:

  El precio son 200 €, **pero hay que tener en cuenta que**:

  - Si el conductor tiene menos de 25 años el precio se incrementa un 6%
  - Si el conductor tiene el carnet menos de 10 años el precio se incrementa un 7%
  - La cobertura contra incendios es de 20€, la cobertura de rotura de lunas es de 30€, pero si se contratan ambas el precio será de 40 € por las dos.

IES HENRI MATISSE
C/ Enric Valor s/n - 46980 Paterna
Telf.961206280- Fax.961206281
46022622@gva.es
www.ieshenrimatisse.es

- **toString(),** que devuelva los datos del seguro con el siguiente aspecto:

```
Año: 2014
Matrícula: 2533-BBC
Incendio: Incluido (o excluido)
Lunas: Incluido (o excluido)
Conductor: Miguel Pérez – 42 años – Carnet de 2005
Precio: 260 €
```

- **compareTo():** Un seguro se considera menor si es de menor año. A igual año, se considera menor si tiene menor matrícula (según el criterio de comparación de Strings).

SeguroRobo

En esta modalidad de seguro queda cubierto el robo del vehículo. Además de toda la información que tiene un Seguro, se conoce la **antigüedad** del vehículo (int). Dicha antigüedad se tendrá en cuenta para calcular el precio del seguro.

Implementa los siguientes métodos (sólo los que se piden):

- **constructor** que recibe los datos necesarios, teniendo en cuenta que los SeguroRobo incluyen siempre la cobertura de rotura de lunas.
- **toString**: Un seguro de robo se mostrará con el siguiente formato

```
Año: 2014
Matrícula: 2533-BBC
Incendio: Incluido (o excluido)
Lunas: Incluido (o excluido)
Conductor: Miguel Pérez – 42 años – Carnet de 2005
Precio: 260 €
Antigüedad del vehiculo: 5 años
ROBO INCLUIDO
```

- **calcularPrecio()**: El precio será el habitual (tal y como se calcula en Seguro) más

  25 € si el vehículo tiene más de 5 años.

  75 € si el vehículo tiene 5 años o menos.

IES HENRI MATISSE
C/ Enric Valor s/n - 46980 Paterna
Telf.961206280- Fax.961206281
46022622@gva.es
www.ieshenrimatisse.es

SeguroTodoRiesgo

Además de toda la información que tiene un SeguroRobo, un seguro a todo riesgo tiene una **franquicia** (double). La franquicia se tendrá en cuenta para calcular el precio del seguro. Cuanto mayor es la franquicia, más barato cuesta el seguro.

Implementa los siguientes métodos (sólo los que se piden):

- **constructor**: que recibe los datos necesarios. La compañia solo permite franquicias que estén entre 100 y 800 €. Si la franquicia indicada al crear el objeto no cumple esta condición, se lanzará la excepción IllegalArgumentException.
- **toString**: Un SeguroTodoRiesgo se mostrará con el siguiente formato

```
Año: 2014
Matrícula: 2533-BBC
Incendio: Incluido (o excluido)
Lunas: Incluido (o excluido)
Conductor: Miguel Pérez – 42 años – Carnet de 2005
Precio: 260 €
Antigüedad del vehiculo: 5 años
ROBO INCLUIDO
Franquicia: 200 €
DAÑOS PROPIOS INCLUIDOS
```

- **calcularPrecio**: El precio será el habitual (tal y como se calcula en la clase SeguroRobo), a lo que añadiremos:
  - 250 € menos la décima parte de la franquicia. Por ejemplo, si la franquicia fuera de 100€ habría que añadir 250 - 10, es decir 240€ al precio habitual del seguro.

IES HENRI MATISSE
C/ Enric Valor s/n - 46980 Paterna
Telf.961206280- Fax.961206281
46022622@gva.es
www.ieshenrimatisse.es

Ejercicio 3: (Este ejercicio usa la clase Seguro del ejercicio 2.)

La clase Presupuestos

Esta clase representa presupuestos de seguro que se han hecho a clientes para lo cual tiene un único atributo, que es un ArrayList de objetos Seguro. El constructor de la calase también está hecho ya.

Se pide implementar los siguientes métodos:

- public void anyadir(Seguro s): Añade el seguro s a la lista s, teniendo en cuenta que no se permiten más de 3 seguros con la misma matrícula:
    - Si ya hay 3 seguros con la misma matrícula que s, se lanzará la excepción IllegalArgumentException.
    - En caso contrario, s se añadirá al final de la lista
- public boolean esCliente (Conductor c): Devuelve true si la lista contiene algún seguro hecho al conductor c y false en caso contrario. Se valorará que no se hagan más iteraciones de las necesarias.
- public double incrementoPrecios(int anyo1, int anyo2): Se quiere saber cuánto han variado de media los precios de los seguros en un año con respecto a otro. Para ello el método devolverá la diferencia entre el precio medio de los seguros de anyo1 y el precio medio de los seguros de anyo2. El método se hará recorriendo una sola vez la lista. Se supone que existen seguros en la lista tanto de un año como de otro.

IES HENRI MATISSE
C/ Enric Valor s/n - 46980 Paterna
Telf.961206280- Fax.961206281
46022622@gva.es
www.ieshenrimatisse.es

Ejercicio4

Disponemos de un ArrayList<Oferta> con ofertas hoteleras. Oferta es una clase definida en el paquete que tiene dos atributos: nombre del hotel y precio de la habitación (ya tiene impementados el constructor y los métodos getter de los dos atributos)

En el ArrayList pueden aparecer varias ofertas de un mismo hotel y queremos averiguar cuál es el mejor precio de cada uno de los hoteles.

Se pide implementar el método buscarMejoresPrecios que, a partir de la lista de ofertas, devuelva un Map en el que la clave sea el nombre del hotel y el valor sea el menor precio con el que aparece el hotel en la lista de ofertas.

Para las ofertas definidas en el método main, el resultado esperado sería el siguiente (no necesariamente en el mismo orden):

```
{Las Arenas=95.0, Parador El Saler=101.0, Melia Plaza Valencia=88.0, Westin
Valencia=86.0}
```

Ejercicio 5

Disponemos de un fichero de texto "nombres.txt" que contiene nombres de personas, que pueden aparecer en mayúsculas o minúsculas. El fichero puede contener nombres duplicados y además está desordenado. Cada nombre está en una línea

a) Se pide, crear un segundo fichero "nombresOrdenados.txt" que contenga los mismos nombres que el fichero original pero sin duplicados y ordenado alfabéticamente

   Al finalizar, se mostrará por pantalla, cuantos nombres han desaparecido del fichero original. Se considerará que un nombre está duplicado sin tener en cuenta mayúsculas o minúsculas. En el fichero nombresOrdenados.txt todos los nombres aparecerán en mayúsculas.

   Si el fichero no existe se mostrará el correspondiente mensaje de error.

Ejemplo de ejecución

nombres.txt	nombresOrdenados.txt
Miguel Angel luis PEDRO miguel pedro angel Luis	ANGEL LUIS MIGUEL MIGUEL ANGEL PEDRO

```
Han desaparecido 2 nombres
```

IES HENRI MATISSE
C/ Enric Valor s/n - 46980 Paterna
Telf.961206280- Fax.961206281
46022622@gva.es
www.ieshenrimatisse.es

b) Modifica el ejercicio para que funcione igualmente cuando algunas líneas contienen números, como en el siguiente ejemplo de ejecución. Los números no se consideran nombres válidos y se ignoran.

nombres.txt	nombresOrdenados.txt
Miguel Angel luis 20 PEDRO miguel 30 pedro 18 angel Luis	ANGEL LUIS MIGUEL MIGUEL ANGEL PEDRO

```
Han desaparecido 2 nombres
```

Ejercicio 6

En la clase Ejercicio6 disponemos de un array que contiene objetos Piscina y objetos camión Cisterna.

Queremos mostrar por pantalla cuántos litros hacen falta para rellenar dichos camionesCisterna y Piscinas.

El mentodo rellenar de Piscina debe devolver la longitud por la anchura, por la altura por 1000.

El método rellenar de CamionCisterna, debe devolver su capacidad.

Se pide:

- Implementer el método rellenar de las clases CamionCisterna y Piscina.
- En la clase Ejercicio6, calcular los litros necesarios para rellenar los elementos del array, añadiendo el/los interfaceces y haciendo los cambios en las clases que sean necesarios.

IES HENRI MATISSE
C/ Enric Valor s/n - 46980 Paterna
Telf.961206280- Fax.961206281
46022622@gva.es
www.ieshenrimatisse.es

1r DAM **Ex. Final – 2020-2021 (07/06/2021)** **Módul: PRG**

Parte pendiente	Puntos	Ejercicios a realizar
EV 2 + EV 3	2,5	Ejercicio 1: Matrices
	1,5	Ejercicio 2: Clase Seguro
	1,5	Ejercicio 2: Clase SeguroRobo
	2,5	Ejercicio 3: Clase Presupuestos
	2	Ejercicio 4: Colecciones ***Ejegir uno de los dos*** Ejercicio 5: Ficheros
SOLO EV 3	1	Ejercicio 2 Clase Seguro (no hacer el método compareTo)
	1,5	Ejercicio 3. Clase SeguroRobo.
	1,5	Ejercicio 3. Clase SeguroTodoRiesgo
	2	Ejercicio 4: Colecciones
	2	Ejercicio 5: Ficheros
	2	Ejercicio 6: Interfaces / Genericidad