EJERCICIO 1: PROGRAMACIÓN ESTRUCTURADA

Debes desarrollar un **programa para la GESTIÓN DE FACTURAS** del Departamento de Informática del IES Alonso de Avellaneda. **Para ello vas a programar todo lo necesario en la clase Ejercicio1.java del paquete util.**

Al iniciar el programa **se debe mostrar un MENÚ con estas opciones** (usa el método **pintaMenu()** de **Ejercicio1.java**):

```
********** EJERCICIO 1 *********
**** MENÚ ****
1. Introducir datos
2. Informe completo
3. Salir
Elige una opción:
```

OPCIÓN 1: INTRODUCIR DATOS

- Por consola hay que **introducir el número de facturas que se van a gestionar.**
    - Damos por hecho que se va a introducir correctamente un número entero positivo.
- Después, por cada factura debemos **introducir 3 gastos añadiendo el importe de cada uno de ellos (puede ser decimal).** Pregunta al usuario y almacena esa información en la estructura de datos que consideres adecuada. Ejemplo de salida por consola:

```
Elige una opción:
1
Introduce el número de facturas:
2
* Introduce los gastos de la factura <1>
        * Gasto <1> :
100
        * Gasto <2> :
50,5
        * Gasto <3> :
30
* Introduce los gastos de la factura <2>
        * Gasto <1> :
20
        * Gasto <2> :
13,75
        * Gasto <3> :
10
```

- Es necesario que se muestre por cada factura y gasto su número para que quede intuitiva la información.
- Siempre habrá tres gastos por factura.
- Si no sabes pedir los gastos por consola inicializa con datos ficticios a mano para poder realizar la opción 2 del menú.

OPCIÓN 2: INFORME COMPLETO

- Si se elige la **opción 2** de menú sin introducir previamente los datos (opción 1) debe salir un mensaje avisando “No has introducido primero los datos” y el menú debe volver a aparecer.

```
Elige una opción:
2
No has introducido primero los datos
**** MENÚ ****
1. Introducir datos
2. Informe completo
3. Salir
Elige una opción:
```

- Obtener **informe completo** con los gastos de cada factura, el importe final de la factura y el importe total de todas las facturas. Por ejemplo:

```
Elige una opción:
2
        Factura (1):
        Gasto (1): 100.0
        Gasto (2): 50.5
        Gasto (3): 30.0
Factura 1, total importe: 180.5 euros.
        Factura (2):
        Gasto (1): 20.0
        Gasto (2): 13.75
        Gasto (3): 10.0
Factura 2, total importe: 43.75 euros.
Importe total: 224.25 euros.
```

  El menú debe volver a aparecer.

OPCIÓN 3:

- El programa mostrará un aviso y se cerrará.

```
Elige una opción:
3
Vas a salir del programa
```

CUALQUIER OTRA OPCIÓN:

- Si se escribe una opción no válida, por ejemplo un 5, saldrá un mensaje avisando y volverá a salir el menú.

```
Elige una opción:
5
Has elegido una opción incorrecta!
**** MENÚ ****
1. Introducir datos
2. Informe completo
3. Salir
Elige una opción:
```

EJERCICIO 2: POO I

En la clase **Ejercicio2.java**, en el método **ejecuta()** hay una array de cadenas con información sobre matrículas de alumnos. Cada cadena está compuesta por:

- nombre del alumno
- su DNI
- su edad

- el número de módulos en los que se ha matriculado
- el ciclo (Grado Medio: GM, Grado Superior: GS).

En el paquete **modelEJ2** hay una clase llamada **Matricula** que mapea toda esa información.

Debes procesar esa información para **crear objetos Matricula** y añadirlos a una **lista dinámica.**
Si no sabes hacerlo, carga la lista con objetos Matricula a mano!!!!

Implementa dos métodos estáticos privados en la clase Ejercicio2 en base a los siguientes requisitos:

- **esRepetida**:
  - En el método ejecuta() tienes un objeto matrícula de prueba. Haz los cambios pertinentes para que deje de dar error su creación.
  - Usa ese objeto como argumento a la función.
  - Debe devolver un booleano si la Matrícula está repetida o no.
  - Pinta el resultado por consola
- **obtenerNombreAlumnoMayorImporte**:
  - Obtiene el nombre del objeto alumno que ha pagado más por la matrícula, teniendo en cuenta que la matrícula de un módulo en GS son 70 euros y en GM son 50 euros.
  - Obligatoriamente debe devolver una cadena con el nombre del alumno. Píntalo por consola.

```
********** EJERCICIO 2 **********
1. La matricula del DNI 11111111B está repetida
2. El nombre del alumno que ha pagado más por la matrícula:Sonia
```

EJERCICIO 3: POO II

Vas a desarrollar la **aplicación de gestión de tributos del Ayuntamiento de Alcalá**, que gestionará **tres tipos de tributos:**

- tributo sobre **BienesInmuebles**
- tributo esto sobre vehículos de tracción mecánica de cuatro ruedas, **VehiculoCuatroRuedas**
- tributo sobre vehículos de tracción mecánica de dos ruedas, **VehiculoDosRuedas**.

Completa la clase Ejercicio3.java y crea las clases pertinentes en base a las siguientes especificaciones y aplicando los pilares del paradigma orientado a objetos.

Un **Tributo** se caracteriza por:

- Su **identificador**: único y consecutivo. La primera solicitud tendrá un valor 3524.
- **NIF** del titular.
- **Fecha pagado**: fecha en la que se ha realizado el pago del tributo. Se completa con la fecha actual en el momento de hacer el pago.

Para **dar de alta un Tributo** en el sistema es necesario especificar únicamente el NIF.
Para **dar de alta cada tipo de tributo** es necesario especificar todos sus atributos.

Para crear la jerarquía de clases y definición debes tener en cuenta:

- **BienesInmuebles**: es necesario conocer los metros cuadrados de la vivienda, y su referencia catastral (cadena de texto que identifica a la vivienda).
- **VehiculoDosRuedas:** es necesario conocer su matrícula y su cilindrada (cm3).
- **VehivuloCuatroRuedas:** es necesario conocer su matrícula y caballos fiscales.

Crear correctamente la jerarquía de clases (model), la definición de sus atributos y constructores. Uso adecuado de getters y setters según especificaciones; La clase TributoDAO no tendrá errores.

Todos los tributos se pagan (tienen ese comportamiento)**.**

- Al pagar **se inicializa la fecha de pago con la fecha actual.**
- **El importe de dicho tributo se calcula de diferente forma dependiendo del tipo de tributo:**
  - **BienesInmuebles:** 10 € por metro cuadrado.
  - **VehiculoDosRuedas**: 0,08 por centímetro cúbico
  - **VehiculoCuatroRuedas**: 4,43 por caballo fiscal.

La clase **TributoDAO** te dará la lista de tributos con los que trabajar.

- **Listar todos los tributos:** se mostrará por consola la información de cada tributo

```
* LISTAR TODOS LOS TRIBUTOS:
BienesInmuebles{Tributo [identificador=3524, NIF=11111111Y, fechapagado=null] metros=75.0, referenciaCatastral=1BV30988764}
BienesInmuebles{Tributo [identificador=3525, NIF=21111111Y, fechapagado=null] metros=85.0, referenciaCatastral=1BV30988765}
BienesInmuebles{Tributo [identificador=3526, NIF=71111111Y, fechapagado=null] metros=95.0, referenciaCatastral=1BV30988766}
BienesInmuebles{Tributo [identificador=3527, NIF=81111111Y, fechapagado=null] metros=100.0, referenciaCatastral=1BV30988767}
VehiculoCuatroRuedas{Tributo [identificador=3528, NIF=31111111Y, fechapagado=null] matricula=0988YD, caballosFiscales=12.3}
VehiculoCuatroRuedas{Tributo [identificador=3529, NIF=41111111Y, fechapagado=null] matricula=0987YZ, caballosFiscales=12.3}
VehiculoDosRuedas{Tributo [identificador=3530, NIF=51111111Y, fechapagado=null] matricula=0987YE, cmcubicos=1340.0}
VehiculoDosRuedas{Tributo [identificador=3531, NIF=61111111Y, fechapagado=null] matricula=0988YF, cmcubicos=1340.0}
```

- **Listar los tributos de bienes inmuebles ordenados de forma natural:**
  - La ordenación natural será por NIF del titular.

```
* LISTAR LOS TRIBUTOS DE BIENES INMUEBLES ORDENADOS DE FORMA NATURAL:
BienesInmuebles{Tributo [identificador=3524, NIF=11111111Y, fechapagado=null] metros=75.0, referenciaCatastral=1BV30988764}
BienesInmuebles{Tributo [identificador=3525, NIF=21111111Y, fechapagado=null] metros=85.0, referenciaCatastral=1BV30988765}
BienesInmuebles{Tributo [identificador=3526, NIF=71111111Y, fechapagado=null] metros=95.0, referenciaCatastral=1BV30988766}
BienesInmuebles{Tributo [identificador=3527, NIF=81111111Y, fechapagado=null] metros=100.0, referenciaCatastral=1BV30988767}
```

- **Listar los tributos de vehículos cuya matrícula contenga “7Y”**

```
* LISTAR LOS TRIBUTOS DE VEHÍCULOS CUYA MATRÍCULA CONTIENE 7Y:
VehiculoCuatroRuedas{Tributo [identificador=3529, NIF=41111111Y, fechapagado=null] matricula=0987YZ, caballosFiscales=12.3}
VehiculoDosRuedas{Tributo [identificador=3530, NIF=51111111Y, fechapagado=null] matricula=0987YE, cmcubicos=1340.0}
```

- **Pagar tributo bienimnueble por referencia catastral:**
  - Si la referencia catastral no existe, debe mostrarse un mensaje por consola.
  - Tras pagar hay que mostrar un mensaje indicando que se ha pagado dicho tributo y el importe del mismo.
  - Antes de hacer el cargo, es importante **comprobar que el impuesto no está pagado previamente**. **Si el impuesto está pagado** se deberá propagar la excepción **ErrorPagoException**.
    Esta excepción propia mostrará el siguiente mensaje:
    “EL TRIBUTO CON IDENTIFICADOR **XXXXX** ESTÁ PAGADO CON FECHA **YYYY-MM-DD**”

  Crea, propaga y captura la excepción correctamente.

```
* PAGAR BIENESINMUEBLES CON REFERENCIA CATASTRAL 1BV66666666
No existe la referencia 1BV66666666

* PAGAR BIENESINMUEBLES CON REFERENCIA CATASTRAL 1BV30988765
tributo!!!BienesInmuebles{Tributo [identificador=3525, NIF=21111111Y, fechapagado=null] metros=85.0, referenciaCatastral=1BV30988765}
Pagado el tributo con referencia catastral 1BV30988765
El importe del mismo:850.0

* VOLVER A INTENTAR PAGAR BIENESINMUEBLES CON REFERENCIA CATASTRAL 1BV30988765
tributo!!!BienesInmuebles{Tributo [identificador=3525, NIF=21111111Y, fechapagado=2024-06-03] metros=85.0, referenciaCatastral=1BV30988765}
********** EXCEPCIÓN PROPIA **********
EL TRIBUTO CON IDENTIFICADOR 3525 ESTÁ PAGADO CON FECHA 2024-06-03
**********
```