# RELACION DE EJERCICIOS 10

## Ejercicio 1

Se pide realizar una clase **Cuenta** con los siguientes métodos

- Un constructor sin ningún argumento que cree una cuenta con saldo 0.
- Un constructor que cree una cuenta con un saldo inicial
- Método get para obtener el saldo de la cuenta.
- Un metódo para sacar una cantidad de la cuenta, no se puede dejar la cuenta en números rojos
- Un método para ingresar una cantidad en la cuenta

Además se debe realizar una clase **ProbandoCuenta** que cree varias objetos Cuenta y pruebe los métodos anteriores

A continuación realizar una clase **CuentaCredito** que herede de la clase Cuenta y que cumpla además:

- Al crear una cuenta de crédito se indica que cantidad de crédito se dispone, es decir en cuanto puede quedar la cuenta en números rojos. Por ejemplo si el crédito es 100 euros la cuenta podrá llegar a tener un saldo igual a –100.
- Inicialmente si no se indica nada, el saldo de la cuenta es 0.
- Inicialmente s i no se indica nada, el crédito es de 100 euros
- Se debe incluir un métodos get y set para el crédito. El crédito nunca puede superar los 300 euros. También habrá que tener en cuenta el saldo actual de la cuenta.
- Se deberá modificar los métodos de sacarDinero para incluir el crédito.

Realizar una clase de prueba MenuCuentaCredito que cree una cuenta de crédito y presente un menú con estas opciones.

1. Ingresar dinero
2. Sacar dinero
3. Mostrar saldo y credito
4. Salir

## Ejercicio 2

Realizar la clase abstracta Figura con el método abstracto area() y las clases Rectángulo, Círculo y Triángulo que hereden de Figura.

Realizar una clase para implementar una Array Aleatorio de Figuras. Tendrá los siguientes métodos

- Un método constructor para crear el array de figuras con el tamaño como argumento. Se deben crear aleatoriamente figuras situadas en los puntos de (0,0) a (99,99) y con longitudes de 1 a 9.
- Un método que imprima para cada figura del array de que tipo es, los valores de sus lados y su área.
- Un método que nos devuelva la figura que tiene el área mayor.

Realizar una clase para probar el array aleatorio, que cree un array aleatorio con 7 figuras, lo imprima e informe de cuál es la figura con área mayor.

**Ejercicio 3**

Se desea desarrollar una aplicación que permita calcular el precio en una empresa de alquiler de vehículos. Cada vehículo se identifica por su matrícula. Los vehículos pueden ser de gama alta, media o baja. La empresa alquila 3 tipos de vehículos: coche, microbús y furgoneta de carga.
El precio base de alquiler del vehículo es de 30 euros al día si es de gama baja, 40 si el de gama media y 50 si es de gama alta. En caso de alquiler de un coche al precio base se le suma la cantidad de 3.5 euros por día si el vehículo es gasolina y 2 euro por día si el vehículo es diesel. En caso de alquiler de un microbús se le añade la cantidad de 5 euros por plaza que disponga el microbús.
En el caso de furgonetas, al precio base se le añade 20 euros * PMA (peso máximo autorizado)

Se debe presentar un menú con las siguientes opciones:

1. Alta de vehículo: Se solicitará el tipo de vehículo y sus datos y lo dará de alta.
2. Cálculo de precio de alquiler: Se solicitará la matrícula del vehículo, el número de días que ha sido alquilado y se mostrará el precio del alquiler
3. Salir

Considerar que la empresa trabajará con un máximo de 200 vehículos.

**Ejercicio 4**

Se va a programar un juego de rol, del cual se nos ha encargado programar en Java parte del esquema de personajes. Para ello se nos han dado las siguientes directrices:

- Tenemos que programar 2 tipos de personajes, los Magos y los Clérigos.
- Todos los personajes cuentan con los siguientes datos:
  - nombre: una cadena.
  - raza: una cadena que puede tomar los valores “humano”, “elfo”, “enano” u “orco”.
  - fuerza: un entero entre 0 y 20
  - inteligencia: un entero entre 0 y 20
  - puntos de vida máximos: un entero entre 0 y 100
  - puntos de vida actuales: un entero entre 0 y puntos de vida máximos
- Además cada tipo de personaje tiene atributos y restricciones específicos que se detallaran en los apartados correspondientes.

**Se pide:**
**Apartado 1:**
Escribe una clase Personaje que reúna los atributos mencionados en el enunciado. Dicha clase debe incluir:

- Un constructor para poder inicializar los atributos (se supone que los puntos de vida actuales son iguales a los máximos al inicializarse)
- Métodos get y set para todos los atributos de la clase
- Método imprime que saque por pantalla los datos del personaje

**Apartado 2:**
Escribe la clase Mago teniendo en cuenta las siguientes restricciones.

- Al crearse, un mago no puede tener en inteligencia un valor menor que 17 ni en fuerza un valor mayor que 15
- Además un mago almacena los nombres de los hechizos que ha memorizado. Un mago sólo puede memorizar a la vez un máximo de 4 hechizos. Impleméntalo como un array y añade los siguientes métodos:

    - aprendeHechizo: que tiene un parámetro de tipo String y que añade un hechizo al array (deberá buscar un hueco libre en el array).
    - lanzaHechizo: que tiene como parámetro un objeto de tipo Personaje que será el personaje sobre el que recae el hechizo y el String correspondiente a un hechizo. Las acciones a tomar serán las de restar 10 de los puntos de vida actuales de dicho personaje y olvidar el hechizo (borrarlo del array).
- Escribe el constructor de la clase (puedes suponer que en el momento de su creación, un mago no conoce ningún hechizo).
- Reescribe el método imprime para que se muestren los nuevos datos incluyendo la lista de hechizos.

**Apartado 3:**
Escribe la clase Clérigo teniendo en cuenta las siguientes restricciones.

- Al crearse, un clérigo no puede tener una fuerza con un valor menor que 18 y una inteligencia con un valor menor que 12 ni mayor que 16 ambos incluidos
- El clérigo reza a un dios para obtener el don de la curación. Por lo tanto se deberá modificar el constructor genérico para aceptar el nombre del dios del cual el clérigo es devoto y así poder almacenarlo.
- Un clérigo tiene el don de curar, por lo tanto, la clase deberá tener un método curar que recibe como parámetro un objeto de tipo Personaje sobre el que recae la acción de curar y que aumenta en 10 los puntos de vida
- Reescribe el método imprime para que muestre además de los datos básicos el nombre del dios.

**Apartado 4:**
Escribe una clase de prueba con un método main en la que se creen 2 magos (A y B) y un clérigo (C) y en el que tengan que realizar las siguientes acciones.

- Imprimir los datos de los tres personajes
- El mago A aprende 2 hechizos.
- El mago B aprende 1 hechizo.
- Imprimir los datos de los magos
- El mago A lanza un hechizo sobre el mago B
- El mago B lanza un hechizo sobre el mago A
- El clérigo cura al mago B
- El mago A lanza un hechizo sobre el mago B
- Imprimir los datos de los tres personajes

**NOTA**: los casos en los que un atributo pueda tomar valores no permitidos contrólalos lanzando la excepción PersonajeException

**Ejercicio 5**

Sobre el ejercicio anterior , realizar un programa principal con el siguiente menú:

1. **Alta de personaje**: Se solicitará si desea crear un clérigo o un mago. Se solicitarán los datos correspondientes y se creará
2. **Mago aprende hechizo**: Se solicitará el nombre del mago y el nombre del hechizo que aprende. Un mago no puede aprender un hechizo que ya tenía aprendido.
3. **Mago lanza hechizo**: Se solicitará el nombre del mago y el nombre del personaje al que quiere lanzar un hechizo. Debe controlarse que no puede lanzarse un hechizo a él mismo. Si al lanzar el hechizo el personaje se queda con menos de cero puntos, el personaje muere por lo que se eliminará.
4. **Clérigo cura al mago**: Se solicitará el nombre del clérigo y el nombre del mago al que quiere curar, y se curará. Un clérigo no puede resucitar a un personaje que ya está muerto
5. **Mostrar el listado de personajes**
6. **Mostrar el listado de personajes ordenados por puntos actuales de mayor a menor**
7. **Salir**

Existirán un máximo de 100 personajes en el juego

**Ejercicio 6**

Dada la siguiente interfaz:

```
public interface CreableEstadisticas
{
	double minimo();
	double maximo();
	double media();
}
```

Modificar la clase ArrayFiguras del ejercicio 2 para que implemente esta interfaz. Los métodos devolverán el area mínima entre todas las figuras, el área máxima y la media de las areas.

Modificar la clase ArrayPersonajes del ejercicio 5 para que implemente también la interfaz. Devolverá los puntos de vida mínimos, puntos de vida máximo, y media de los puntos de vida de los personajes.

Probar los métodos anteriores.