Actividad: Coches.

Vamos a acceder desde Java, utilizando la API JDBC, a un SGBD, en nuestro caso MySQL, para crear y trabajar con una base de datos relacional.

Apartado 1. Creación de la base de datos.
Utilizando un contenedor Docker o Docker Compose, ejecutar el script BDCoches.txt sobre MySQL y mostrar los resultados obtenidos.

Apartado 2. Operaciones sobre la BD desde Java con JDBC.
La conexión a la base de datos y la consulta de la misma se desarrollará en una clase java llamada AccesoDatos. Las pruebas se realizarán desde otra clase PruebaAccesoDatos que solo contendrá una función main desde donde se llamará a todas las funciones de AccesoDatos.
La clase AccesoDatos deberá implementar los siguientes métodos:

- public void abrirConexion(), para conectarse a la BD DatosCoches.
- public void cerrarConexion(), para cerrar la conexión una vez terminadas las consultas.
- public void mostrarDatosCoches(), para obtener los registros de la tabla Coches ordenados según el precio de mayor a menor.
- public void modificarCoche(String matricula, int precio), para modificar la tabla Coches pasando la matrícula del coche que se quiere modificar y el nuevo precio.
- public void borrarCoche(String matricula), para borrar el coche cuya matricula se pasa como argumento.
- public void insertarCoche(String matricula, String marca, int precio, String dni), para insertar un nuevo registro en la tabla Coches, pasándole como argumento la matrícula, marca, precio del coche y dni del propietario.
- public void insertarPropietario(String dni, String nombre, int edad), para insertar un registro en la tabla Propietarios. Los argumentos del método son el DNI, nombre y edad.

Apartado 3. Clase que prueba el acceso a datos.
Para probar todas las funciones que se han hecho en la clase AccesoDatos, desarrollar la clase PruebaAccesoDatos con una función main:

```
public static void main (String[] args) throws SQLException{
        AccesoDatos AD = new AccesoDatos();
        AD.abrirConexion();
        AD.mostrarDatosCoches();
        AD.modificarCoche("BA-3333", 5000);
        AD.borrarCoche("MA");
        AD.insertarCoche("AA-0005", "Ford", 4500, "1A");
        AD.insertarPropietario("X25", "Jose", 54);
        AD.cerrarConexion();
}
```

Apartado 4. Funciones adicionales.
Hasta ahora no se han verificado las restricciones que se exigen cuando se accede a una base de datos con tablas relacionadas entre si. A continuación deben implementardos unos métodos para ello:

- Método que dado el dni del propietario, liste sus datos y los coches que posee.
- Modificar el método de insertar coches para que verifique que el dni del propietario ya existe en la tabla propietarios, si no existe, que no permita la inserción del nuevo coche.
- Método que permita borrar de la BD a un propietario (borrando también los coches de este propietario).

Suban al aula virtual un archivo pdf con el enlace a los archivos de los proyectos de NetBeans. El nombre del archivo debe formarse con tu nombre y el número de actividad, por ejemplo: “Javier. Actividad 1.03.pdf”

Esta actividad se calificará de 0 a 10.