Ejercicio de base de datos.

Crear la siguiente base de datos **libreria**:

```
CREATE DATABASE libreria;
USE libreria;

CREATE TABLE libros(
codigo varchar(4),
autor varchar(50),
titulo varchar(100),
precio varchar(10),
stock varchar(10)
);

CREATE TABLE ventas(
codigoventa varchar(4),
fechaventa varchar(100),
importetotal varchar(10),
librosadquiridos varchar(100)
);
```

Antes de empezar cargar nuestro **driver** mysql-connector-java-5.1.22-bin dentro de la biblioteca de nuestro proyecto.

Crear la siguiente clase java llamada **Libro** con los atributos: codigo, autor, titulo, precio y stock todos ellos de texto y privados. Incluir sus métodos get y set para los atributos y un constructor para los cinco atributos.

Crear la siguiente clase java llamada **Venta** con los atributos: codigoVenta, fechaVenta e importeTotal todos ellos de texto y privados. Incluir sus métodos get y set para los atributos.

Crear el fichero principal **Gestion**.java en él importaremos la clase java.sql.* necesaria para el manejo de base de datos la clase java.io.* para guardar en ficheos y la clase java.util.ArrayList para un ArrayList (vector de tamaño dinámico).
En nuestro **main** lo primero será crear objeto conexión de la clase gestion llamar al método **abrirConexión** donde nos pedirá la base de datos y todo el código necesario, cuando tengamos conexión con la bbdd nos aparecerá el siguiente menú:

Gestión libreria

1- Insertar Libro
2- Borrar Libro
3- Mostrar Libros
4- Buscar Libro
5- Cambiar Stock
6- Insertar Venta
7- Crear fichero venta
8- Salir

Cada opción en un switch con su respectivo método.

Método insertarLibro:

Crearemos un objeto Libro llamado por ejemplo libroNuevo, rellenaremos sus cinco valores mediante sout y los set correspondientes.
Cuando ya este rellenado pasamos a meter ese libroNuevo en la bd, primero la gestión de errores (cuando trabajamos con ficheros o bd) con la orden *try* dentro de esta comenzamos creando un Statement y ejecutando un update sobre la bd pasandole la String para insertar.

Método borrarLibro:

Guardamos en un String el codigo del libro a borrar.
Dentro de un *try* creamos el statement y ejecutamos el update con la sentencia SQL de eliminación del libro.

Método mostrarLibros:

Mas sencillo simplemente crear el Statement y el Resulset y mostrar en bucle por pantalla los datos de la bd recuperando los cinco campos con getString(“nombrecolumna”).

Método buscarLibro:

Para buscarLibro al principio pedimos un Codigo del libro lo guardamos y después dentro de un *try* creamos el Statement.
Hemos de crear un Resulset que ejecute la consulta deseada y posteriormente en un bucle while mostrar cada una de las filas por pantalla recuperando los cinco campos con getString(“nombrecolumna”).

Método cambiarStock:

Pedir el codigo del libro y el stock a modificar, posteriormente en el try crear el Statement y pasarle el método executeUpdate para lanzar el SQL necesario.

Método insertarVenta:
Método largo por un lado rellena un **ArrayList** con los libros que se han vendido y por otro lado **inserta en la bd** en la tabla ventas sus valores.

Crear objeto Ventanueva de la clase Venta, pedir por pantalla los datos codigoVenta fechaVenta e importeTotal. Después crear siguiente menú:

Libros adquiridos

1- Añadir libro
0- Terminar venta

Opción 1:
Crear objeto **libroventa** de la clase libro para la venta, pedir el código libro que se desea vender, despues dentro de un *try* crear una consulta para recuperar toda la información de ese código libro y

gracias a un bucle recorrer toda la solución del SQL y meter el resultado en el objeto libro creado antes.
Ahora en libroventa ya tenemos toda la información del libro a vender a continuación insertar ese libro dentro del ArrayList con: librosvector.add(libroventa);

En una variable String guardar el historico de los códigos de los libros adquiridos a la venta.

Opción 0: Para salir con un break y ya no añadir más libros.

Al acabar este método insertar en la tabla ventas los valores necesarios que son: el CodigoVenta, la FechaVenta, el ImporteTotal (estos tres los recogeremos del objeto ventanueva y los gets asociados) el último valor a registrar en la tabla son los librosadquiridos guardado en la variable librosstring.

Método CrearFichero:

Crear archivo Ventas.txt. para cada elemento del ArrayList

Método cerrarConexion:

En un try ya que nos obliga al trabajar con bd cerrar la conexión con .close() y gestionar su excepción.

Método abrirConexion:

Crear un objeto Connection para crear la conexión con la bd, comenzar un *try* para la gestión de errores en db y pedir por pantalla en nombre de la bd a trabajar, cargar el Driver correspondiente con: Class.forName("com.mysql.jdbc.Driver");
Realizar la conexión con nuestra base de datos guardada en la string bbdd:
con = java.sql.DriverManager.getConnection("jdbc:mysql://localhost/" + bbdd, "root", "");
Devolver mediante un return con.