


Evaluación: 1 – UT2 Conexiones a bases de datos relacionales	Fecha: 22/11/2024	Calificación	
Ciclo: DAM	Curso: 2ºB		
Módulo: ACDA – Acceso a Datos	Dpto.: INFORMÁTICA		
Nombre y apellidos:			

En este programa vamos a acceder a una tabla en una base de datos relacional de manera que podamos hacer operaciones CRUD básicas y búsquedas parametrizadas. Para ello tendremos una estructura de clases con diferentes cometidos (conexión, modelo, DAO, servicios) que tendremos que ir completando o modificando para el funcionamiento correcto del programa según lo solicitado. El programa preguntará primero si queremos recargar los datos de la tabla (para poder hacer las pruebas sin preocupaciones) y seguidamente mostrará un menú que nos permitirá hacer operaciones CRUD básicas. En el transcurso del examen tendremos que añadir una opción nueva que nos permita hacer una búsqueda por un campo diferente a la clave principal.

Para completar el examen deberás:

- Programar los apartados **TODO** numerados en el proyecto. Existen etiquetas **TODO START** y **TODO END** para cada uno de los apartados solicitados, ya sean para modificar el código ya escrito o para escribir nuevo código. **La totalidad** del código que escribáis deberá quedar entre estas etiquetas. En caso de modificar algo fuera de las etiquetas **TODO** el examen podría no ser corregido.
- No añadir ninguna clase, método o atributo de clase nuevos (no son necesarios). De ser así **la calificación del examen podría ser un 0**.
- Entregar un proyecto funcional. **Si no compila** la calificación del examen podría ser un **0**.
- Igualmente, si hay **errores en la ejecución sin manejar de manera específica** (incluyendo la entrada de cualquier dato por teclado) la calificación podría ser **0**.
- En caso de no conseguir ejecutar el código escrito sin errores, puedes dejarlo comentado pero el programa tendrá que ser totalmente funcional (aunque no haga lo que se pide) para que se revise.
- Únicamente podrá estar abierto en el ordenador **IntelliJ con el examen**, **Workbench** para la consulta de la base de datos y los archivos **pdf** de apuntes y presentaciones proporcionados por el profesorado. **En caso de haber cualquier otra cosa abierta (incluyendo otros archivos o proyectos diferentes al examen) la calificación del examen será 0**.
- Se valorará adicionalmente el uso sistemático (no casual) de **logger** tanto para los errores como para la información relevante (hasta 0.5 puntos)

PARTE 1:

1. **Asegúrate de haber creado la base de datos que usará el programa con el script proporcionado. Revisa y comprende** la estructura y tipos de datos de la tabla *Libros* creada en la base de datos. Revisa la clase *Libro* del modelo y entiende qué datos almacenan los atributos y su correspondencia en la base de datos. Lee bien todos los apartados de este documento ya que en el archivo java puede no estar bien reflejado todo lo solicitado.
2. Clase constants.LibroConstants: define los valores a asignar a las constantes situadas entre las etiquetas **TODO**. (0.25p)
3. Clase db.connection.MySQLConnection: modifica el código situado entre las etiquetas **TODO** de manera que el programa pueda conectarse a la base de datos. (0.25p)
4. Clase db.dao.LibroDAO: modifica o reescribe el código entre las etiquetas **TODO** del método *actualizar* de manera que **no se puedan producir inyecciones de SQL** (según lo visto en el tema). (1.5p)

5. Clase *db.dao.LibroDAO*: completa el método *obtenerPorAutor*. El método debe buscar en la base de datos coincidencias parciales o totales en la columna *autor* con el texto proporcionado como parámetro de entrada. Devuelve una lista de *Libro* (que estará vacía si no hay coincidencias). Deberás usar *PreparedStatement* de manera obligatoria, si no lo usas no se tomará en cuenta lo escrito. (1.5p)

Ampliación: una vez el método funcione (y hayas completado el resto del examen), intenta ampliarlo de manera que, si el parámetro de entrada es *null*, devuelva aquellos libros cuyo autor sea *null*. Esto te va a exigir hacer dos consultas diferenciadas y gestionarlas. (0.5p)

PARTE 2:

6. Clase *db.service.LibroService*: modifica o reescribe el código entre las etiquetas *TODO* del método *actualizar* para que las operaciones de actualización de registros se hagan mediante transacciones. (1.5p)
7. Clase *db.service.LibroService*: escribe un método de servicio que valide y haga uso del método DAO *obtenerPorAutor*. Devolverá una lista de *Libro* con las coincidencias de la búsqueda o una lista vacía si no ha habido ninguna. (1.5p)
8. Clase *db.service.LibroService*: completa el método *comprobarLibro*. Deberás escribir entre las etiquetas *TODO* el código que haga la validación para el atributo *titulo*. Además de las restricciones que nos ponga la propia base de datos, no admitiremos valores en blanco o vacíos. Si la longitud es mayor de la que permite la base de datos haremos uso de los métodos auxiliares de la clase *utils.GeneralUtils* para recortar la longitud. (0.75p)
9. Clase *db.service.LibroService*: completa el método *comprobarLibro*. Deberás escribir entre las etiquetas *TODO* el código que haga la validación para el atributo *autor*. Si la longitud es mayor de la que permite la base de datos haremos uso de los métodos auxiliares de la clase *utils.GeneralUtils* para recortar la longitud. (0.75p)

PARTE 3:

10. Clase *Main*: modifica el código del método *opcionesUsuario* para que incluya una nueva operación de búsqueda por autor, que se ejecutará con el método *buscarAutor*. (0.25p)
11. Clase *Main*: modifica el código del método *mostrarMenu* para que muestre en pantalla una nueva opción de hacer búsquedas por autor de manera similar a las que ya aparecen. (0.25p)
12. Clase *Main*: completa el código entre las etiquetas *TODO* del método *buscarAutor*. Este método debe hacer uso del método de servicio *obtenerPorAutor* para recuperar una lista de libros cuyo autor tenga coincidencias con lo introducido por teclado, o una lista vacía si no hay coincidencias. Seguidamente se imprimirá en pantalla (*System.out*) los libros o se notificará si la lista está vacía. (1.5p)

Rúbrica de corrección (calificación máxima 10 en cualquier caso)

Paso (pts)	Corrección	Pts.
2 (0.25)		
3 (0.25)		
4 (1.5)		
5 (1.5)		
6 (1.5)		
7 (1.5)		
8 (0.75)		
9 (0.75)		
10 (0.25)		
11 (0.25)		
12 (1.5)		
Ampl. 5 (0.5)		
Logs (0.5)		