EJERCICIO 1

Crea una clase Aula con un atributo número (Integer) y que tenga una lista de Alumno.

La clase Alumno tendrá:

- Nombre (String)
- DNI (String)
- Nota (BigDecimal)
- Fecha de nacimiento (LocalDate)

Crea get y set en las dos clases.

Añade los siguientes métodos en la clase Aula. Si necesitas crear algún método auxiliar en la clase Aula o Alumno, puedes hacerlo.

EJERCICIOS PARA PRACTICAR LA BÚSQUEDA:

a) obtenerAlumnoMayor() que devuelve el alumno de mayor edad que haya aprobado.
b) obtenerAlumnoNombreMasLargo() que devuelve el alumno con el nombre de mayor tamaño

EJERCICIOS PARA PRACTICAR LA ACUMULACIÓN:

c) obtenerSumaNotas() que devuelve un BigDecimal redondeado a 1 decimal con la suma de todas las notas de la clase.
d) obtenerEdadMediaMeses() que devuelve la edad media de los alumnos en meses como un entero.

Aplica variaciones de estos métodos según te vaya indicando el profesor:

- Condicionantes distintos
- Lanzar excepciones
- Borrado de elementos con iterador
- Obtener lista o mapa filtrando resultados

EJERCICIO 2

Cambia la clase Aula para que en lugar de tener una lista de alumnos tenga un mapa de alumnos de tipo String → Alumno, donde la key es el DNI de cada alumno. Repite todos los ejercicios.

EJERCICIO 3

En un aula, cuando está vacía, hay 20000 litros de oxígeno. Crea un método calculaOxigenoRestanteEnUnaHora() que devuelva el oxigeno que queda en el aula después de una hora teniendo en cuenta que:

- Los alumnos que son mayores de edad consumen 30 litros en una hora.
- Los alumnos que son menores de edad consumen 40 litros en una hora.
- Los alumnos no aprobados consumen un plus de 10 litros la hora.

EJERCICIO 4

Crea una tabla llamada ALUMNOS con estas columnas:

- DNI → VARCHAR (50) → PK
- NOMBRE → VARCHAR (100)
- CALIFICACION → DECIMAL(6,2)
- FECHA → DATE

Construye una clase AlumnosService en un paquete "service". Crea estos métodos:

- insertarAlumno() que recibe un Alumno y lo inserta en BBDD. Si hay algún error lanza una excepción llamara ErrorBBDDException con un mensaje. Puedes ignorar los errores al cerrar statement y connection.
- InsertarAlumnos() que recibe una lista de alumnos y los inserta todos en BBDD. Si hay algún error al insertar alguno, muestra un mensaje por consola y continúa intentando insertar el siguiente. Si hay algún otro error al acceder a la BBDD, lanzará ErrorBBDDException. Los errores al cerrar conexión y statement los puedes ignorar.
- insertarAlumnosCompleto() que recibe una lista de alumnos y los inserta todos en BBDD. Si hay algún error al insertar alguno, debe deshacer la transacción completa y no insertar ninguno. Y luego lanzar ErrorBBDDException. Si hay un error al deshacer la transacción, muestra un mensaje de error por consola. Los errores al cerrar conexión y statement los puedes ignorar.

EJERCICIO 5

Repite los dos últimos métodos recibiendo un mapa de alumnos con clave DNI y valor Alumno.

EJERCICIO 6

Repite todos los métodos anteriores (del ejercicio 4 y 5) pero con un UPDATE. Es decir, los métodos serán actualizarAlumno(), actualizarAlumnos() y actualizarAlumnosCompleto()

EJERCICIO 7

Añade a la clase estos métodos:

- consultarAlumno() que recibe un dni y devuelve un Alumno. Si el alumno no existe, lanzará AlumnoNotFoundException. Si hay algún error lanza una excepción llamara ErrorBBDDException con un mensaje. Puedes ignorar los errores al cerrar statement y connection.
- consultarAlumnos() que recibe una notaDesde y una notaHasta y devuelve una lista de alumnos que tengan una nota comprendida entre ambas. Si no hay ninguno, devolverá una lista vacía. Trata los errores igual que en el caso anterior.
- consultarAlumnosMapa() que sea igual que el anterior, pero devuelva un mapa con clave DNI y valor el Alumno.

EJERCICIO 8

Crea una clase App con un método main que solicite los datos de un alumno al usuario, teniendo en cuenta que:

- El nombre no puede ser vacío. Si lo es, vuelve a solicitarlo.
- El DNI tiene que tener exactamente 9 caracteres. Si no es así, vuelve a solicitarlo.
- La nota tiene que estar comprendida entre 0 y 10. Será un BigDecimal. Si no es así, vuelve a solicitarla. Si el usuario introduce datos que no son un número decimal, captura el error, muestra un mensaje, y vuelve a solicitarlo.
- La fecha de nacimiento tiene que ser anterior a hoy. Si no es así, vuelve a solicitarla. Si el usuario introduce una fecha no válida, captura el error, muestra un mensaje, y vuelve a solicitarla.

EJERCICIO 9

Volviendo a la clase Aula, crea un método que sea calcularCosteTotalMatriculas() que devuelva un BigDecimal con la suma de todo lo que costará las matriculas de todos los alumnos. Para ello, ten en cuenta que:

- El precio normal de la matrícula de cada alumno es de 3000 euros
- Si el alumno no ha aprobado, tendrá que pagar el doble.
- Si el alumno es menor de edad, tendrá un descuento del 15%

EJERCICIO 10

Crea un método que sea obtenerAlumnosDatosIncorrectos() que devuelva una Lista con todos los alumnos cuyos datos no estén bien. Para que los datos de un alumno sean correctos, se tiene que cumplir:

- El nombre no puede ser vacío ni sólo tener espacios en blanco
- El DNI tiene que tener longitud 9 y no contener ningún guion
- La nota tiene que estar entre 0 y 10
- La fecha de nacimiento será anterior a hoy

EJERCICIO 11

Hemos prometido regalar unos bombones a los alumnos por cada punto que obtengan en el examen. Crea un método que sea calcularBombonesTotales() que devuelva un número entero con el total de bombones que vamos a regalar. Ten en cuenta que:

- Por cada punto obtenido en la nota, regalamos un bombón.
- Si has aprobado, tienes un bombón adicional
- Si has sacado un 10, tienes dos bombones adicionales más.
- Los alumnos menores de edad no reciben bombones porque no tenemos permiso de sus padres para que puedan comer chocolate.

EJERCICIO 12

Modifica el método insertarAlumnosCompleto() del ej 4 para que sólo inserte aquellos que estén aprobados. Tendrá que devolver el número de alumnos insertados en total.

EJERCICIO 13

Modifica el método insertarAlumnos() del ej 4 para devuelva una lista con todos los alumnos que finalmente se han insertado sin errores.

EJERCICIO 14

Añade a los métodos del ej 7 estos otros métodos:

- consultarAlumnosAdultosA() que devuelva un Set de alumnos con todos los alumnos mayores de edad. Hazlo consultando todos los alumnos y luego filtrando los resultados al recorrer el ResultSet para sólo añadir a la lista los mayores de edad.
- consultarAlumnosAdultosB() que devuelva un Set de alumnos con todos los alumnos mayores de edad (igual que antes). En este caso, hazlo añadiendo un filtro al SQL en el WHERE. Tendrás que poner que la fecha de nacimiento sea menor a una fecha determinada que tendrás que calcular previamente.
- consultarAlumnosPorFecha() que devuelve una lista de alumnos y recibe dos fechas (fecha inicio y fecha fin). Tiene que devolver todos los alumnos que haya en base de datos cuyas fechas de nacimiento estén comprendidas entre ambas.

EJERCICIO 15

Volviendo a la clase Aula, crea un método que reciba una letra (String) y borre todos los alumnos cuyo DNI tengan dicha letra. Llama al método borrarAlumnosLetraDni()

EJERCICIO 16

Crea en la clase Aula un método que devuelva una lista de alumnos y recibe un año (Integer). Tendrá que devolver en la lista todos los alumnos que hayan nacido en el año indicado. Llama al método obtenerAlumnosAño()

EJERCICIO 17

Crea en la clase Aula un método que devuelva un Integer y reciba un BigDecimal que será una nota. El método tiene que devolver cuántos alumnos hay con una nota igual o mayor a la recibida por parámetro. Llama al método cuentaAlumnosNota()