RECUPERATORIO INTEGRADOR

Leé completo y con cuidado el enunciado. Pensá bien la estrategia de resolución antes de comenzar el desarrollo de lo que te solicitan. El objetivo de este examen es evaluar la correcta aplicación de los conceptos y técnicas vistos hasta el momento:

- *Correcta implementación de constructores.*
- *Modularización reutilizable y mantenible con uso de métodos con correcta parametrización y correcto encapsulamiento, publicando setters y getters sólo cuando corresponda.*
- *Manejo de clases y colecciones*
- *Manejo de Asociación, Agregación y Composición.*
- *Manejo de relaciones jerárquicas. Herencia, Clases Abstractas e Interfaces.*
- *Manejo de Arreglos uni y bi dimensionales.*
- *Manejo de TDAs – Pilas, Colas y Listas ordenadas*
- *Importación y Exportación de proyectos Java desde Eclipse*

Enunciado

Una empresa que gestiona depósitos nos pide un prototipo de software para la gestión de cada uno de los mismos.

En esta primera etapa nos piden concentrarnos en el depósito que poseen, que almacena productos de tres tipos solamente.

El depósito tiene una estantería rectangular con 3 niveles y 4 estantes en cada nivel. En cada uno de los niveles se almacenará sólo un tipo de producto, los tipos de producto son Pelota de tenis, Bolsos de Tenis y Raquetas de tenis. Todos y cada uno de los Estantes tendrán cierta cantidad de ubicaciones (su profundidad) que será igual para cada uno de ellos, y no podrá ser menor a 2 ni mayor a 10 lugares. Cada estante puede ser accedido solo por su lado frontal, es decir que el último producto almacenado allí, será el primero que se pueda sacar.

Todos productos a almacenar tendrían como información un id, entero que no puede ser 0 ni negativo, una marca, del tipo texto que no podrá ser nula ni vacía, estos productos deberán tener la capacidad de mostrar su información por consola.

Las raquetas tendrán, además, el tamaño de su aro (entero), que no puede ser menor a 93 ni mayor a 120 pulgadas cuadradas. Los bolsos poseerán un modelo, que no puede ser ni nulo ni vacío. Y por último las pelotas serán para ser utilizadas en una superficie (Césped, Polvo de ladrillo o Rápida).

Tanto el depósito como los estantes deberán ser del tipo Depositante, por lo que deberán poder depositar elementos y retirarlos basados en el ID del elemento a retirar.

Además de la estantería, el depósito deberá llevar registro de los productos alojados, para poder indicar fácilmente, si un producto con un determinado ID está depositado o no.

RECUPERATORIO INTEGRADOR

Para aprobar el examen se deberá realizar lo siguiente:

1. Implementar las clases **Producto**, **Raqueta, Pelota** y **Bolso** aplicando las validaciones indicadas.
2. Implementar el método mostrar() definido en la interfaz Mostrable, donde considere pertinente en base a la salida esperada.
3. Completar la clase **Deposito**, agregando los atributos y métodos necesarios para cumplir con el funcionamiento
    a. Constructor deberá recibir por parámetro la profundidad de la estantería.
    b. Método ***`void depositar(Producto p) throws RuntimeException`*** deberá intentar almacenar el producto recibido en la primera ubicación de la fila que corresponda a su tipo, que tenga lugar. Si no hay lugar, deberá arrojarse una excepción indicando el problema.
    c. Método ***`boolean productoDepositado(int idProducto)`*** deberá indicar si hay depositado un producto con el identificador recibido.
    d. Método ***`Producto retirarPorId(Integer id)`*** deberá sacar el producto de la estantería y del registro. Si no se encuentra se debe devolver null.
4. Completar la clase Estante, agregando los atributos y métodos necesarios.
    a. Constructor deberá recibir por parámetro la capacidad máxima del estante.
    b. Método ***`depositar(Producto p) throws RuntimeException`*** deberá intentar agregar el producto en el estante, si no hay lugar deberá arrojar una excepción para indicar la situación. El producto deberá ser agregado de forma tal que se pueda retirar primero el último que se agregó.
    c. Método ***Producto retirarPorId(Integer id)*** debe retirar un producto en base al ID indicado. Si el producto no está deberá devolver null. El estante debe quedar ordenado como estaba antes de la operación.
5. Completar la gestión de errores en la clase Test. Solo se debe modificar lo que se considere pertinente para la gestión de errores, no modificar el resto de la clase.

Para probar el funcionamiento del programa se dispone de una clase de prueba “Test” que contiene la creación del depósito y dispone de un lote de pruebas de productos que serán almacenados.

NOTA: Esta clase no deberá ser modificada por los alumnos, salvo la gestión de errores.

RECUPERATORIO INTEGRADOR

Respetando los datos que ya están cargados la salida debería ser la siguiente:

```
Depositando productos

Bolso depositado id: 10
Raqueta depositada id: 11
Pelota depositada id: 12

Error depositando bolso: La marca no puede ser nula ni vacía
Error depositando raqueta: El id no puede ser menor a 1

Bolso depositado id: 14
Raqueta depositada id: 15
Pelota depositada id: 16

Error depositando bolso: Modelo de bolso inválido.
Error depositando raqueta: Tamaño de aro inválido.
Error depositando pelota: La superficie no puede ser nula

Pelota depositada id: 20
Pelota depositada id: 21
Pelota depositada id: 22
Pelota depositada id: 23
Pelota depositada id: 24
Pelota depositada id: 25
Pelota depositada id: 26
Pelota depositada id: 27
Pelota depositada id: 28
Pelota depositada id: 29

Error depositando pelota: No se pudo depositar el producto.

Bolso depositado id: 31
Fin de los depósitos

El producto id 55 NO está depositado
El producto id 23 está depositado
Se comienza a retirar elementos

Se retiró el producto id: 12
La pelota para superficie POLVO_LADRILLO es de la marca Wilson y tiene un id 12

No se pudo retirar producto id: 13

Se retiró el producto id: 28
La pelota para superficie CESPED es de la marca Wimbledon y tiene un id 28

Fin del retiro de elementos
```