

## Segundo Recuperatorio - Julio 2025

Para el siguiente ejercicio, se debe realizar el diagrama UML y escribir el código Java. Se deben desarrollar los métodos indicados por las funcionalidades y también cualquier otro que fuera necesario para cumplir con las mismas, en las clases en que corresponda.

**Se debe entregar el diagrama de clases UML en formato .uxf y el proyecto Java exportado de Netbeans en formato .zip**

No se aceptarán archivos .pdf ni tampoco archivos de imagenes. Los archivos deben ser subidos al campus en el link de la actividad “segundo recuperatorio”. No se recibirán archivos que se envíen por correo electrónico. La resolución no debe incluir técnicas no enseñadas durante el curso.

**IMPORTANTE: NO SE TOMARAN EN CUENTA ARCHIVOS ENTREGADOS EN OTROS FORMATOS.**

La Tienda UtnShop se dedica a la comercialización de artículos electrodomésticos y nos pide una aplicación para optimizar la gestión de sus depósitos y sus artículos. Tiene cinco Depósitos, divididos en dos categorías: Comunes y Frágiles. Los depósitos comunes pueden albergar hasta 10 artículos, mientras que los frágiles disponen los artículos en ubicaciones en dos estanterías, de 10 ubicaciones cada una (por lo tanto pueden guardar hasta 20 artículos como máximo).

Los depósitos se crean automáticamente al momento de crearse la Tienda: serán tres de tipo común y los restantes de tipo frágil.

Cada depósito debe poder manejar el almacenamiento según su tipo. Esto significa que debe poder ingresar un artículo (guardar un artículo) y poder informar el lugar que tiene disponible o ninguno si no lo tiene. La forma en que realice estas tareas dependerá de la estructura de almacenamiento de cada uno.

De cada artículo se conoce su código (un String, que no puede ser nulo ni vacío), el nivel de resistencia de su embalaje (puede ser ALTARESISTENCIA,BAJARESISTENCIA), y cuál es la empresa fabricante. Las fábricas que trabajan con la empresa son: SanPung, Thilips, Poblex y Elgil. Ni la fábrica ni la resistencia del embalaje pueden ser nulas.

La tienda guarda un registro de todos los artículos que ha recibido de las fábricas durante la última semana. Estos artículos son los que están pendientes de ingresar a depósito y no pueden ser nulos.

Cada semana, los artículos pendientes deberán ser ingresados a alguno de los depósitos.

**El programa debe resolver las siguientes funcionalidades:**

- Constructor de tienda: debe crear todos los depósitos que tiene la tienda (mencionados más arriba).
- IngresarArticulosPendientes: la tienda debe agregar el artículo recibido como parámetro a su colección de pendientes.
- Mostrar la fábrica que más artículos pendientes de ingreso tiene: debe retornar cuál es la empresa con mayor cantidad total de artículos en la colección de pendientes..
- Ingresar todos los artículos a depósito: debe tomar todos los artículos que están pendientes de ingreso e ir guardándolos en el primer depósito que tenga lugar disponible. Debe tenerse en cuenta la resistencia del embalaje del artículo para ubicarlo en un depósito (por ej., si es de baja resistencia debe estar en un depósito para frágiles). Si no hubiera lugar en ninguno se deberá tratar esa situación con una excepción. Tener en cuenta que cada tipo de depósito tiene distinta estructura de almacenamiento. Cada artículo que pueda ser guardado en un depósito deberá quitarse de la colección de pendientes.
- Gestionar las siguientes excepciones en donde corresponda.
  - que no hubiera lugar para ingresar un artículo a depósito.
  - que no se pudieran agregar un artículo a la lista de pendientes.

**Se provee la clase PruebaTienda con la creación del objeto Tienda y la creación de sus artículos pendientes ya codificada. También contiene los métodos necesarios para probar las clases diseñadas.**

**Tener en cuenta que el proyecto provisto y sus clases están INCOMPLETOS. Se deben completar clases, métodos y todo el manejo de excepciones.**