TAREA INTEGRADORA 2

MercadoLibre ha contratado sus servicios para desarrollar una aplicación que permita la venta online de su tienda virtual. Su tarea es crear un programa que permita ingresar el inventario de la tienda virtual y permita la búsqueda y eliminación de productos. Además, el programa le debe permitir registrar los pedidos que hacen los usuarios de la tienda.

INGRESO Y ELIMINACIÓN DE DATOS

Para los productos, la aplicación debe permitir al usuario ingresar información sobre cada producto, incluyendo su nombre, descripción, precio, cantidad disponible, categoría y número de veces comprado.

Considere categorías como: Libros, Electrónica, Ropa y accesorios, Alimentos y bebidas, Papelería, Deportes, Productos de belleza y cuidado personal, Juguetes y juegos

Para los pedidos, la aplicación debe permitir al usuario ingresar información como nombre del comprador, lista de productos, precio total y fecha de la compra. Al momento de generar un pedido, tenga en cuenta que las cantidades del inventario disminuyen.

El usuario puede aumentar la cantidad de cada producto ya registrado.

Proponga un menú adecuado en cada situación y estructúrelo correctamente para que el usuario pueda agregar todos los productos y pedidos que quiera.

BUSCADOR

El programa debe contar con un buscador de productos y un buscador de pedidos.

El buscador de productos debe permitir buscar productos por *nombre*, *precio*, *categoría* y *número de veces comprado*.

El buscador de pedidos permite buscar por *nombre de comprador*, *precio total* y *fecha de compra*

Los buscadores deben utilizar el algoritmo de búsqueda binaria para encontrar coincidencias en los registros de productos y deben mostrar los resultados por consola.

Además, la aplicación debe permitir búsquedas por rango para los atributos numéricos, como el precio, la cantidad disponible o el número de veces comprado. El usuario debe poder especificar un valor mínimo y máximo para buscar los productos dentro de ese rango. También debe permitir búsquedas por intervalo para los atributos de tipo string, como el nombre, donde el usuario pueda definir una letra o prefijo de inicio y una letra o prefijo final para buscar productos que se encuentren en ese rango alfabético.

Antes de mostrar los resultados, el buscador debe permitir al usuario escoger el orden (ascendente o descendente) de los datos que se muestran y la variable de ordenamiento.

Por ejemplo, el usuario puede buscar productos por rango de precios. El usuario introduce 0 como límite mínimo y 10000 como límite máximo. El buscador reúne los resultados y antes de mostrarlos, pregunta al usuario con qué variable desea hacer el ordenamiento y si lo hace de forma ascendente o descendente. El usuario escoge el número de veces comprado y escoge orden descendente. El resultado dará un reporte de todos los productos que cuestan menos de 10000 ordenados de menor número de veces comprado hasta mayor número de veces comprado.

PERSISTENCIA

Para garantizar la integridad de los datos, la aplicación debe contar con persistencia de datos mediante serialización JSON. Debe ser capaz de almacenar la información de los productos en archivos para que los datos estén disponibles incluso después de que se cierre el programa.

EXCEPCIONES

Finalmente, el programa debe manejar excepciones para evitar estados inesperados durante el ingreso y la búsqueda de productos. Debe proporcionar mensajes de error informativos para guiar al usuario en caso de que se produzca un error inesperado.

CONDICIONES

El proyecto DEBE ser desarrollado usando TDD.

Además, use un repositorio desde el día 1. A lo largo del desarrollo debe hacer al menos 10 commits repartidos equi-temporalmente. En cada uno de los 10 commits reporte 3 indicadores de calidad sencillos en el readme del repositorio. La idea es que se pueda hacer un seguimiento de la evolución de los indicadores

Indicadores de calidad

Densidad de errores-fallos = total de fallos / total de pruebas
Confiabilidad = 1 - densidad de fallos
Completitud = casos de prueba / total funcionalidades
* Este último indicador se espera que sea un número mayor que 5 al final de la implementación del producto.

Entregas

1. **Requerimientos + casos de prueba de funcionalidades (15%)**
   Especifique los requerimientos a partir de lo que aparece en el enunciado de la tarea integradora. A partir de los requerimientos, escriba al menos 2 pruebas por funcionalidad. Pueden ser test negativos o positivos que compongan una prueba completa de la aplicación. RECUERDE QUE SÓLO DEBE ESCRIBIR PRUEBAS DEL MODELO DE LA APLICACIÓN. No incluya pruebas que involucren a la consola o la capa de interacción con el usuario.
   Reporte la lista inicial de pruebas en el formato de este documento.
   Entrega: Final de la semana 10.

2. **Evolución del programa y seguimiento de Indicadores (5%)**
   En este punto se medirá la evolución de su repositorio. Para esto vaya reportando progresivamente los indicadores de calidad en 10 commits del programa. Escoja 10 versiones equi-temporales. Escriba en el readme una sección de indicadores en el que irá acumulando los indicadores versión tras versión.

   …
   `Indicadores`

```
Iteración 1
   Densidad de errores-fallos = 0.2012
   Confiabilidad = 0.7988
   Completitud = 2.32

Iteración 2
   Densidad de errores-fallos = 0.4043
   Confiabilidad = 0.5957
   Completitud = 3.01
…
```

3. **Implementación (65%) y diseño UML (15%) completos**
   Esta entrega debe tener la implementación funcionando perfectamente acorde a los requerimientos. Anexe el diseño, esta vez en su versión final.
   Entrega: Final de semana 13