Universidad de Valladolid

Departamento de Informática

Programación Orientada a Objetos

Segunda entrega de Prácticas

Curso 2021/22

1. Supuesto práctico

Una vez que la empresa ***VendingCo*** ha podido probar la corrección de la funcionalidad que ha sido desarrollada para el anterior supuesto, ha decidido aplicarla, con unos ligeros cambios, a todo su negocio en España; asimismo, para poder diferenciarse de sus competidoras, va a optar por diversificar la oferta de productos, por lo que, todos estos cambios van a tener que ser actualizados en la ampliación de esta solución.

2. Funcionalidad a proveer

En primer lugar, la empresa quiere gestionar su negocio por toda España. En cada provincia hay una sede de ***VendingCo*** y serán cada una de las sedes las que gestionarán las máquinas vending de la provincia.

Por otro lado, para ampliar la oferta de las máquinas vending, se van a vender packs. Los packs serán agrupaciones de varios productos que irá determinando la empresa. Los productos al ser agrupados en un pack podrán comprarse conjuntamente y si se compran de esta forma el precio de venta tendrá un 20 % de descuento respecto de lo que costaría haber comprado todos los productos que forman el pack individualmente.

En la máquina de vending podrán venderse entonces tanto productos individuales como packs creados por la empresa que gestiona la máquina. Se desea que el tratamiento desde la máquina de vending sea homogéneo cuando se venda un producto individual como cuando se venda un pack.

Para ello se realizará una abstracción a partir de la clase Producto ya realizada en la práctica 1. Llamaremos a esta abstracción `Vendible`. Dicha abstracción será común a Producto y a una nueva clase que llamaremos `Pack`.

2.1. Modificaciones

Recordamos que esta práctica no consiste en la implementación de la interfaz de usuario de este sistema sino del conjunto de clases que controlan el sistema utilizado para esta gestión. En ningún caso se trata de clases que interaccionen con los usuarios.

2.1.1. VendingSystem

La clase `VendingSystem` gestiona todo el sistema. Necesita saber qué ocurre en cada una de las sedes con las máquinas vending. La pasada funcionalidad propia de `VendingSystem` pasará a la clase `VendingCity`.

Universidad de **Valladolid.** Dpto. de Informática. Edificio de Tecnologías de la Información y las Telecomunicaciones, Campus Miguel Delibes. 47011 Valladolid. Tfno. 983.423000/Fax 983.423671.
e-mail:marga@infor.uva.es

La nueva funcionalidad de `VendingSystem` será:

- añadir una nueva sede,
- eliminar una sede,
- a partir del identificador de una sede, conocer el número de las máquinas vending,
- obtener la lista de las máquinas vending de una sede,
- conocer el número de provincias que se gestionan,
- obtener los nombres de todas las provincias donde hay una sede,
- obtener una lista de la cantidad de máquinas vending que se gestionan en cada una de las provincias.

2.1.2. VendingCity

Cada clase `VendingCity` se identifica por un código de provincia, y almacena el nombre de esa provincia y las máquinas que se encuentran diseminadas en esa provincia. La funcionalidad de esta clase es la adaptación de la funcionalidad de la clase `VendingSystem` de la práctica 1.

2.1.3. VendingMachine

Esta clase ahora va a almacenar Vendibles porque son una abstracción de `Pack`s y `Product`os.
Su funcionalidad es similar.

2.1.4. Vendible

La clase `Vendible` es una abstracción. Se podrá saber como mínimo su nombre, su precio y el identificador único que manejará la empresa. En el caso de los Productos este identificador podrá estar dado por el UPC.

2.1.5. Pack

En la clase Pack se realizará una inicialización basada en un array de Producto (Producto()). Debe garantizarse que un pack tenga siempre un mínimo de 2 productos, que nunca serán productos repetidos.

De un pack, además de poder conocer su nombre, su precio y su identificador único (como de cualquier elemento Vendible), deberá poder conocerse la cantidad de productos que forman el pack, si un producto dado forma parte del pack, así como saber qué productos en concreto forman dicho pack. El precio del pack se calculará con un descuento del 20 % a la suma de los precios de los productos contenidos en el pack. Deberá garantizarse que si el precio de algunos de los productos del pack cambia, el precio del pack también.

Se deberá poder gestionar el pack añadiendo y eliminando elementos siempre que se respeten los detalles anteriores. En ningún caso se permitirá tener packs formados por otros packs.

2.1.6. Product

Esta clase no va a cambiar.

2.2. Clases

Se espera que las clases que forman el proyecto Eclipse de la entrega contengan , como mínimo, las siguientes clases: `VendingSystem`, `VendingCity`, `VendingMachine`, una clase abstracta `Vendible`, una clase `Pack` y una clase `Product`.

Las clases de test serán las siguientes: `VendingSystemTest`, `VendingMachineTest`, `VendibleTest` y `PackTest`. En el caso de `VendingMachineTest` se recomienda reutilizar los test realizados en la práctica 1, mejorándolos y adaptándolos a la nueva estructura de VendingMachine.

Es necesario aclarar que en presencia de herencia, los métodos deben probarse en cada contexto. Es decir, si un método está definido en una clase y heredado en otras, deberá probarse en cada uno de los herederos pues cada contexto es diferente. Deberán probarse el uso de los objetos polimórficamente. Es decir, cuando un objeto Pack o Producto se usa desde una entidad Vendible.

Si la solución presentada está basada en alguna otra clase adicional a las mencionadas, cada clase del proyecto debe venir acompañada de su correspondiente clase de prueba implementada mediante JUnit 4 y nombrada con el mismo esquema.

3. Condiciones de entrega

- La entrega consistirá en un único archivo `.zip`.
- El archivo contendrá el proyecto Eclipse compatible con la versión Eclipse 2020-06.
- El archivo y el proyecto debe llamarse `entrega1-idAlumno1-idAlumno2`, donde `idAlumno` se refiere al identificador de la cuenta de laboratorio de cada alumno y residir en un directorio del mismo nombre.

  Ejemplo: `entrega1-javper-margar`
- El proyecto debe compilar.
- La entrega se realizará mediante la subida del archivo zip a una tarea habilitada al respecto en el aula virtual. La entrega debe hacerse una sola vez por equipo. El equipo es responsable de decidir cuál de sus integrantes es el encargado de subir la práctica.
- El límite para la entrega de la práctica se establece en las **23:55 del 22 de diciembre de 2021**
- **No se admitirán entregas que incumplan estas condiciones**.
- En caso de incumplimiento de las condiciones anteriores, se considerará la práctica como **no presentada**.
- Es necesario que el código cumpla las convenciones de código **Java**[1].
- Es necesario documentar las clases mediante comentarios **JavaDoc**[2].
- Es deseable que cada archivo `.java` contenido en la entrega tenga en la cabecera (comentarios **JavaDoc**) el nombre de los autores (mediante la etiqueta `@author`). Para indicar el nombre, se preferirá el identificador de la cuenta de laboratorio de cada alumno en lugar de su nombre completo.

[1] http://www.oracle.com/technetwork/java/codeconvtoc-136057.html
[2] http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html

4. Referencias

- https://www.bic-code.org/bic-codes/
- https://www.stocklogistic.com/codigos-contenedor-maritimo/
- https://en.wikipedia.org/wiki/ISO_6346#Check_Digit