Práctica 4: Máquina expendedora.

1. El problema

Se quiere realizar un programa para simular el funcionamiento de una máquina expendedora que sea capaz de vender varios productos.

2. Objetivos

- La implementación de clases “de negocio” (o “de datos”), en contraposición a las clases “de librería” o “de programa”.
- Creación y uso de excepciones de usuario. Hasta ahora se han capturado y lanzado excepciones pertenecientes a clases que ya se encuentran en Java. En esta práctica se crearán y usarán nuevas clases de excepción.
- Implementación de programas en los que se encuentran separados la capa de negocio y la capa de presentación. La clase Expendedora será capaz realizar las operaciones típicas de una máquina expendedora, pero no interactuará con el usuario. El interfaz de usuario lo proporcionará una clase distinta. El proyecto sigue el patrón “Modelo-Vista-Controlador” (MVC), que separa en clases distintas el interfaz de usuario (view), las clases que responden a las acciones del usuario (controller) y las reglas de negocio del problema que se resuelve (model).
- Implementación de programas con interfaz gráfico de usuario (GUI). El programa a través del cual el usuario manejará la expendedora no será en modo texto, sino que tendrá un interfaz gráfico.

3. Fases de la elaboración de la práctica.

Para realizar la práctica se seguirán los pasos que se resumen a continuación:

1. Observar la estructura del proyecto.
2. Crear nuevas clases de Excepción necesarias para el funcionamiento de la expendedora.
3. Implementar la clase Expendedora.
4. Probar la clase Expendedora usando el programa de test que se proporciona.
5. Probar la clase Expendedora usando el programa InterfazExpendedoraTexto
6. Implementar un interfaz gráfico para la expendedora.

4. Observar la estructura del proyecto

Junto con el enunciado se te ha entregado un proyecto de JavaFX que tiene los siguientes paquetes:

- model: paquete que contendrá las clases que implementan las reglas de negocio de la expendedora, en concreto, la clase Expendedora y tres clases de excepción (una de ellas ya está creada y las otras dos se crearán en el paso siguiente de la práctica).
- view: paquete que contendrá el archivo fxml que se creará para el interfaz gráfico de usuario.
- controller: Cuando se desarrolle el interfaz gráfico del programa, este paquete contendrá la clase controladora correspondiente al archivo fxml anterior.
- test: contiene dos programas de test con los que probar la clase Expendedora diseñada.

5. Crear nuevas clases de excepción.

Como verás más adelante, la Expendedora lanzará tres tipos de excepciones ante situaciones de error: CreditoInsuficienteException, StockInsuficienteException y CambioInsuficienteException. La primera de ellas ya está creada en el paquete model. Debes crear las otras dos.

Las excepciones son clases prácticamente triviales, derivadas de la clase Exception, muy sencillas de crear. Crealas a imagen y semejanza de la que ya hay

6. Implementación de la clase Expendedora

6.1. Funcionamiento de una expendedora

La clase Expendedora soporta las operaciones típicas que se realizan sobre una máquina expendedora sencilla, que es capaz de dispensar varios productos.

El funcionamiento de una expendedora, a grandes rasgos, es el siguiente:

1. El cliente introduce dinero en la máquina. Al dinero introducido lo llamaremos **crédito**.
2. Pulsa un botón para comprar el producto deseado.
3. Si hay stock del artículo seleccionado, la máquina dispensa el artículo elegido y devuelve el importe sobrante (diferencia entre el crédito introducido y el precio del artículo).

Durante el proceso se pueden producir diversas incidencias, como por ejemplo, que el cliente no haya introducido suficiente crédito para comprar el producto, que no quede producto o que no haya cambio suficiente para la devolución. En cualquiera de estos casos, la venta no se realiza y el cliente tiene la opción de solicitar la devolución del crédito sin realizar la compra.

6.2. La clase Expendedora

En primer lugar, indicar que la clase Expendedora no debe realizar ninguna interacción con el usuario: no solicitará ningún dato ni mostrará información por pantalla.

Atributos (privados)

- credito: Cantidad de dinero (en euros) introducida por el comprador. Las modificaciones de este atributo se deben hacer de forma que tenga dos decimales.
- cambioDisponible: Cambio del que dispone la máquina. El cambio disponible se reduce cada vez que se devuelve al cliente la diferencia entre el crédito introducido y el precio del producto comprado. **El cambio nunca se verá incrementado por las compras de los clientes**: .
- importeVentas: Representa la suma de las ventas realizadas por la máquina (en euros). Se ve incrementada con cada nueva compra.
- Tres arrays para almacenar la información de los productos que la máquina vende. En concreto, el nombre de cada producto, su precio y su stock.
    - nombreProductos: array de String
    - precioProductos: array de double
    - stockProductos: array de int

Constantes

- **CREDITOMAXIMO**: Es el crédito máximo que admite la máquina. La expendedora no permitirá que se le introduzca dinero por más importe que este valor. Se inicializará con el valor 10.

Métodos

- **Constructor**: *public Expendedora (double cambioDisponible, String[] nombreProductos, double[] precioProducto, int[] stockProducto).* Crea la expendedora inicializandola con los datos que ser reciben como parámetro:
    - El método deberá lanzar IllegalArgumentException si los tres arrays no tienen el mismo tamaño.
    - El crédito y el importeVentas se inicializarán a cero.
    - Los tres atributos array se usarán para inicializar los nombres, precios y stocks de los productos que dispensa la expendedora.
- **Consultores**: Se crearán los siguientes métodos consultores **unicamente**:
    - getCredito()
    - getCambioDisponible()
    - getImporteVentas()
    - getNumeroProductos(): Devuelve el número de productos que vende la máquina
    - getNombre(int i): Devuelve el nombre del producto i
    - getPrecio(int i): Devuelve el precio del producto i
    - getStock(int i): Devuelve el stock del producto i

- **Modificadores**: **No se implementarán modificadores.** Se considera que los atributos de la máquina solo van a cambiar por operaciones derivadas de su funcionamiento (introducción de dinero, compras, devoluciones. ...), por lo que **no** proporcionamos modificadores públicos
- **Otros métodos**
  - *public String* ***toString****()* Devuelve un String con la información de la máquina y de los productos que vende, de la forma:

```
Credito           :   9.70 euros
Cambio            :  25.10 euros
Importe de ventas: 115.00 euros
Productos: 2
      Agua 1L – 0.70 euros - 5 uds.
      Cola    – 1.00 euros - 3 uds.
```

  - *public void* ***anyadirDinero*** *(double importe)* Representa la operación mediante la cual el cliente añade dinero (crédito) a la máquina. Esta operación incrementa el crédito introducido por el cliente en el importe indicado como parámetro. El credito no tiene que sobrepasar el valor de la constante CREDITOMAXIMO. Si al introducir el dinero que se indica como parámetro el crédito fuese a superar el CREDITOMAXIMO, la operación no se realizaría (sin mostrar ningún mensaje ni lanzar ninguna excepción).
  - *public double* ***devolverCredito****()* Representa la operación mediante la cual el cliente solicita la devolución del crédito introducido sin realizar la compra. El método devuelve cuánto dinero se retorna al cliente.
  - *public double* ***comprar*** *(int i) throws CreditoInsuficienteException, CambioInsuficienteException, StockInsuficienteException.* Representa la operación mediante la cual el cliente ordena la compra de un producto.

    El parámetro i indica qué producto es el que se quiere comprar. Es decir, qué posición, dentro de los arrays nombreProductos, precioProductos y stockProductos, ocupa el producto que se quiere comprar. Se supone que estará entre 0 y nombreProductos.length - 1

    El método devuelve la cantidad de dinero que se devuelve al cliente, es decir, cuánto le sobra de su compra.

    Si no se produce ninguna situación inesperada, se reducirá el stock del producto, se decrementará el cambio disponible, se pondrá el crédito a cero y se aumentará el importe de ventas.

    Si la venta no es posible se lanzará la excepción correspondiente a la situación que impide completar la venta. La excepción se creará con un mensaje de error que describa la situación de error.

7. Prueba con la expendedora con la clase de Test

Ejecuta la clase TestExpendedora. Si la clase Expendedora se ha implementado correctamente la ejecución no mostrará ningún error. **Nota:** *Es difícil tener en cuenta todas los posibles errores que se pueden cometer al implementar la clase, por lo que es posible que el programa de test no refleje errores y la expendedora no esté completamente bien*

8. Prueba con la clase InterfazExpendedoraTexto

Se trata de un programa en modo texto que crea un objeto Expendedora y permite al usuario interactuar con él. Ejecútalo y observa si la información que aparece es coherente según las operaciones realizadas por el usuario.

Estudia esta clase, porque te puede ayudar a implementar el interfaz gráfico de la expendedora del punto siguiente.

9. Crear un interfaz para la expendedora en modo gráfico

El interfaz gráfico debe cumplir los siguientes requisitos:

- Creará una expendedora de varios productos (entre 4 y 6) con los stocks, nombres y precios que el alumno/a quiera. El número de productos será fijo, no se podrá cambiar a través del programa.

- El interfaz gráfico mostrará el estado de la expendedora: el crédito introducido, el cambio que le queda, el importe de las ventas.

- Para introducir el dinero, se usarán botones que corresponderán a distintas monedas: 2 €, 1€, 0.50 €, etc … Cada vez que se pulse en uno de estos botones se añadirá el importe correspondiente al crédito de la expendedora.

- Habrá un botón por cada producto que vende la máquina. El usuario pulsará el botón correspondiente a un producto para comprarlo. En el texto del botón, se mostrará el nombre del producto, el precio y el stock del producto correspondiente. **Estos tres datos no se deben poner fijos en el interfaz sino que se deben tomar de la máquina expendedora a través de los correspondientes métodos getter.**

- Habrá un botón para que el usuario solicite la devolución del crédito sin comprar.

- Cuando la compra no se pueda realizar, se mostrará un mensaje al usuario indicándole cual es el problema (crédito insuficiente, stock insuficiente, …)

- Cuando la compra se realice con éxito, se mostrará al usuario un mensaje indicándole cual es el cambio que le corresponde.

- Cuando el usuario solicite la devolución del crédito sin comprar, se mostrará un mensaje al usuario indicándole cuanto dinero se le devuelve.

- Para mostrar los mensajes descritos anteriormente, se utilizará la clase Alert de JavaFX. En el siguiente enlace tienes un tutorial con ejemplos de uso: https://code.makery.ch/blog/javafx-dialogs-official/

- También puedes investigar cómo añadir imágenes a los botones.