Máquina De Dulces

(POO Java)

Para iniciar el desarrollo del proyecto se deben crear los paquetes componentes, maquina y test como se detalla a continuación:

En el paquete **componentes** se debe crear una clase llamada **Producto** con las siguientes características:

Atributos	• **Nombre:** Es el nombre del producto • **Precio:** Es el precio del producto en número decimal • **Código:** Es un conjunto de caracteres que identifican al producto
Métodos	• **incrementarPrecio:** recibe como parámetro un porcentaje de incremento de tipo entero, no tiene retorno. Calcula el porcentaje de incremento y le suma al precio actual. • **disminuirPrecio**: recibe un valor de descuento de tipo decimal, no retorna nada. Resta el valor de descuento al precio actual.
Constructores	• Constructor que reciba 3 parámetros correspondiente a los atributos de **Producto** y que asigne sus valores a los atributos.

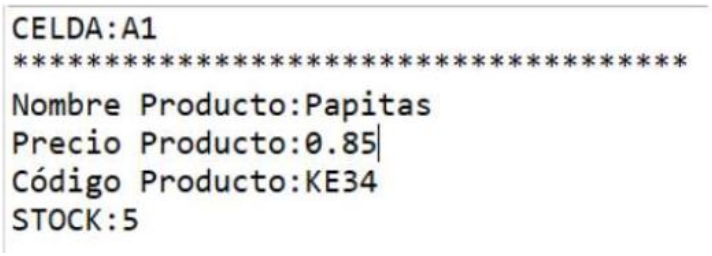
En el paquete **test** crear una clase llamada **TestProducto** que permita validar la clase **Producto** y sus métodos. A continuación, se muestra el resultado esperado:

```
Código:KE34
Nombre:Papitas
Precio:0.85
************************************
Nuevo Precio:0.9
Precio incrementado:1.35
Precio incrementado:1.0
```

En el paquete **componentes** se debe crear una clase llamada **Celda** con las siguientes características:

Atributos	• **Producto:** De tipo producto • **Stock:** De tipo entero • **Código:** Es un conjunto de caracteres
Métodos	• **ingresarProducto:** recibe como parámetro el Producto y el stock inicial, no retorna nada. Coloca los valores en los atributos producto y stock respectivamente.
Constructores	• Constructor que reciba 1 parámetros correspondiente al atributo **Código** de la celda y que asigne su valor al atributo correspondiente.

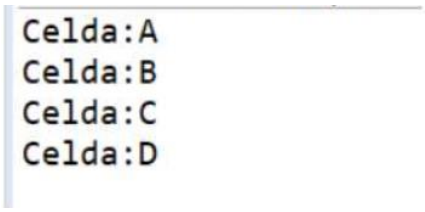
En el paquete **test** crear una clase llamada **TestContenidoCelda** que permita validar la clase **Celda** y su método implementado. A continuación, se muestra el resultado esperado:



En el paquete **maquina** se debe crear una clase llamada **MaquinaDulces** con las siguientes características:

Atributos	• **Celdas:** De tipo ArrayList<Celda> • **Saldo:** De tipo double
Métodos	• **agregarCelda:** recibe el código de una celda. Instancia una celda con dicho código y la agrega a la lista. • **mostrarConfiguracion:** no recibe parámetros y no tiene retorno. Imprime en consola los códigos de todas las celdas que tenga la máquina en ese momento.
Constructores	• Configuración por defecto

En el paquete **test** crear una clase llamada **TestMaquinaDulces** que permita validar la clase **MaquinaDulces** y sus métodos. A continuación, se muestra el resultado esperado:



Una vez comprobado el funcionamiento de la clase **MaquinaDulces,** procederemos a implementar un gran conjunto de métodos que representarán las acciones con las que va a disponer la máquina.

Métodos	Test

buscarCelda: recibe un código de celda y retorna la celda correspondiente a dicho código, si no existe retorna null.	Crear una clase llamada **TestBuscarCelda** que presente el siguiente resultado:
	`Celda:A Stock:0 Sin Producto asignado`
	`Celda:B Stock:4 Producto:KE34 Precio:0.85`
	`Celda:C Stock:0 Sin Producto asignado`
	`Celda:D Stock:5 Producto:BDCR Precio:2.54`

buscarProductoEnCelda: recibe un código de celda y retorna el producto guardado en esa celda, si no existe retorna null.	Crear una clase llamada **TestBuscarProductoEnCelda** que contenga 4 celdas (A, B, C, D), debe registrar un producto con asignando sus atributos y su respectiva celda. Emplear el método **buscarProductoEnCelda** con el código de la celda asignada, y seguidamente emplearlo para una celda inexistente. Resultado esperado:
	`Producto encontrado:Papitas`
	`Producto encontrado:null`

consultarPrecio: recibe el código de una celda y retorna el precio de un Producto guardado en dicha celda.	Crear una clase llamada **TestConsultarPrecio** que contenga productos registrados con su respectiva celda. Ejecutar el método **consultarPrecio** en uno de los productos y corrobore el resultado.
buscarCeldaProducto: recibe el código de un producto y retorna la celda que lo contiene. Si no encuentra retorna null.	Crear una clase llamada **TestBuscarCeldaProducto** que contenga productos registrados con su respectiva celda. Ejecutar el método **buscarCeldaProducto** en uno de los productos y corrobore el resultado.

incrementarProductos: recibe el código de un producto y la cantidad de ítems a incrementar, no retorna nada. Invoca al método **buscarCeldaProducto** usando el código del producto que recibe. Guarda el retorno de **buscarCeldaProducto** en una variable llamada **celdaEncontrada.** En la **celdaEncontrada** agregar el número de ítems al stock actual de productos en la celda.	Crear una clase llamada **TestIncrementarProductos** que contenga 4 celdas (A, B, C, D), con productos registrados con su respectiva celda. Invocar el método **mostrarProductos** antes y después de ejecutar el método **incrementarProductos**. Resultado esperado:
	`Celda:A Stock:0 Sin Producto asignado`
	`Celda:B Stock:4 Producto:Papitas Precio:0.85`
	`Celda:C Stock:0 Sin Producto asignado`
	`Celda:D Stock:5 Producto:Gatorade Precio:2.54`
	`-----------Luego de incrementar---------`
	`Celda:A Stock:0 Sin Producto asignado`
	`Celda:B Stock:10 Producto:Papitas Precio:0.85`
	`Celda:C Stock:0 Sin Producto asignado`
	`Celda:D Stock:5 Producto:Gatorade Precio:2.54`

vender: recibe el código de una celda, no retorna nada. Busca la celda y disminuye su stock en una unidad. Obtiene el precio del producto y suma ese valor al saldo actual de la máquina. Invoca al método **mostrarProductos** para imprimir el saldo de la máquina.	Crear una clase llamada **TestVender** que contenga productos registrados con su respectiva celda. Ejecutar el método **vender** en dos de los productos y corrobore el resultado de la variable Saldo y el stock de los productos.
venderConCambio: recibe el código de la celda y el valor ingresado por el cliente en la máquina, retorna el cambio que debe dar la máquina. Busca la celda, resta una unidad al stock del producto, modifica el saldo y retorna la devuelta que debe entregar al cliente.	Crear una clase llamada **TestVenderConCambio** que contenga productos registrados con su respectiva celda. Ejecutar el método **venderConCambio** en uno de los productos y corrobore el resultado de la variable Saldo, el stock de los productos y el valor de cambio que se debe entregar al cliente.