FACULTAD DE INFORMATICA
CÁTEDRA INTRODUCCIÓN POO
TRABAJO PRACTICO FINAL

Una tienda de indumentaria deportiva desea registrar y sistematizar las ventas de los productos que comercializa. Para ello guarda información de cada uno de sus productos como: precio,talle, color, marca y cantidad stock; y toda la información de la venta realizada: fecha, la colección de productos y el cliente. Las ventas de los productos pueden ser por crédito, débito o efectivo. Inicialmente y basados en el modelo relacional informado por el responsable de diseño y desarrollo de la aplicación, el Mapeo Objeto Relacional cuenta con las siguientes clases: ***Tienda, Producto, Item, Venta y su jerarquía***. En cada una de las clases implementar los métodos básicos para buscar, ingresar, actualizar y buscar los datos en un motor de base de Datos MySQL utilizando la clase BaseDatos.php proporcionada por la cátedra. Además implementar:

- Método constructor que recibe como parámetros los valores iniciales para los atributos definidos en cada clase.
- Los métodos de acceso de cada uno de los atributos de la clase.
- Redefinir el método **_toString** para que retorne la información de los atributos de la clase.

Las clases principales del ***M**apeo **O**bjeto **R**elacional* tienen una clase a nivel de transacciones de la aplicación que implementan las operaciones que permiten las altas, bajas y modificación de los datos además de la implementación de procedimientos complejos.

- Implementar el ABM (**A**lta, **B**aja, **M**odificación) de los productos de la tienda. Además opciones de búsqueda de productos: por nombre, por código de barra, por stock inferior a un valor enviado por parámetro.

En la clase ***Producto*** se registra la siguiente información: precio, código barra, nombre, marca, color, descripción y cantidad en stock. Cada vez que la tienda vende un producto se decrementa su stock teniendo en cuenta la cantidad de productos vendidos en la operación.

- Implementar el método ***actualizarStock*** que recibe por parámetro una cantidad y actualiza el valor del stock del producto según corresponda. Si el valor recibido por parámetro es >0, entonces se incrementa el stock y si el calor es <0 se decrementa el stock del producto.

Para cada ***Venta*** se registra la siguiente información: f*echa, cliente, código de la venta y la colección de items.* Como se mencionó, las ventas pueden ser: *Efectivo, Débito o Crédito*. El importe parcial de la venta se corresponde con la suma de los importes de los items por su cantidad. Si la venta es al **contado** este importe no varia, si es por **tarjeta de débito** tienen las mismas características de una operación al contado, salvo que además se registra si fueron realizadas con una tarjeta de débito de la red Link o de la red Banelco y el banco emisor de la tarjeta. Las operaciones con **tarjeta de crédito**, también cuentan con las características de una operación al contado, pero registran adicionalmente la siguiente información: la cantidad de cuotas, nombre del titular de la tarjeta, número de tarjeta y empresa *(Visa,Mastecard)*.
En la clase **TVenta** implementar el método:

- ***importeParcialVenta*** que retorna el importe de la venta en base a la colección de items de la venta
- ***darImpFinalVenta*** es una venta en **efectivo** se aplica un 10% de descuento al importe parcial de la venta; si la operación se realizó con tarjeta de **débito** no se incluye descuento; y, si la operación se realizó con tarjeta de **crédito** el importe total tiene un incremento del 10%, si el cliente paga en 3 o más cuotas, caso contrario no tiene descuento ni incremento. Redefinir el método cuando sea necesario.

Por cada I***tem*** de la venta se registra el producto y la cantidad de ejemplares del producto que se venden. En la clase **TItem** implementar el método ***darImporteItem*** que retorna el importe parcial del item basado en la cantidad por el precio unitario.
En la clase ***Tienda*** se registra la siguiente información: nombre, dirección, teléfono la colección de productos y la colección de ventas realizadas. Implementar en la clase **Ttienda** los siguientes métodos:

- ***registarVenta*** que recibe 3 parámetros. El primer parámetro es un arreglo asociativo con las siguientes claves: "*unProducto"* (referencia a un objeto producto) y "*cantidad*" (cantidad de ejemplares del producto que desea venderse). El segundo parámetro es el tipo de la venta: **E:** *Efectivo* **D:** *Débito*, **C:** *Credito.* Finalmente el tercer parámetro es un arreglo asociativo con la información propia del tipo de venta. El método crea un nuevo objeto venta según el parámetro recibido y retorna el importe final de la venta que debe abonar el cliente según el tipo de venta. ***Nota****: no se requiere controlar el stock el método supone que el arreglo de productos cumple con las condiciones para su venta y que además deja el stock correctamente si la venta fue exitosa.*
- ***realizarVenta*** que recibe por parámetro un arreglo asociativo con los productos a vender con las siguientes claves:*"codigoBarra"* (código barra correspondiente a un producto) y "*cantidad*" (cantidad de ejemplares del producto que desea venderse), información del cliente y la forma de pago. El procedimiento debe buscar los productos según el código de barra, verificar el stock disponible y realizar el registro de la venta en caso de ser posible. El procedimiento debe retornar un objeto Venta con los ítem correspondientes a aquellos producto que pudo vender. En la implementación del método deben utilizarse los siguientes métodos: ***buscarProducto , incorporarProducto, actualizarStock*** y ***registarVenta.***
- ***ventaMayorImporte*** *que recibe por parámetro un tipo de venta y retorna una referencia a la venta con mayor importe realizada por la tienda. Puede utilizar la función* ***is_a****($obj,'Clase') que retorna true si $obj es una instancia de la clase 'Clase'* o false en caso contrario.
- ***VentaMayorImporteXTipoVenta*** retorna un arreglo asociativo con las siguientes claves: efectivo, débito y crédito donde cada posición del arreglo contiene una referencia al objeto venta con mayor importe.

Implementar un script ***appTienda*** el cual invoca a todos los métodos implementados en las clases transaccionales a partir de un menu de usuario.