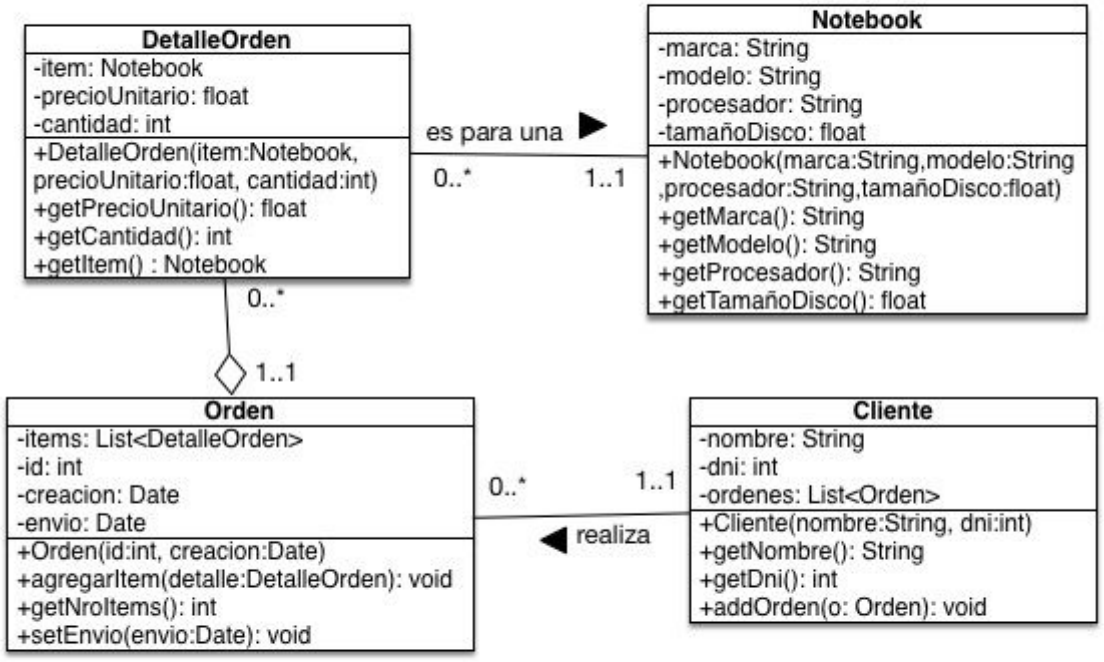
Examen 111Mil – Venta de Notebooks - Solución

Ana es fanática de la tecnología. Hace dos años decidió abrir su propio negocio de informática llamado "La gran Manzana". El negocio se enfoca principalmente en la venta de notebooks a trave internet. Dada la creciente demanda de notebooks, Ana necesita desarrollar un sistema que la ayude en su negocio.
Como sabe que cursamos el programa 111Mil se puso en contacto con nosotros para que la ayudemos a construir el sistema, que en esta primera etapa del proyecto deberá administrar las órdenes de compra de cada cliente.

Ejercicio 1. Implementar desde el diagrama de clases

Dado que el encargado es experto en modelado de software, ha creado un diagrama de clases UML para indicarnos lo que se desea. Otros programadores comenzaron con la codificación pero nos piden que la completemos (solo la clase Notebook está completa).




```
public class Cliente {
   private String nombre;
   private int dni;
   private List<Orden> ordenenes;

   public Cliente(String nombre, int dni) {
      this.nombre=nombre;
      this.dni=dni;
      ordenes=new ArrayList<>();
   }
   public void addOrden(Orden o){
      ordenes.add(o);
   }
   public String getNombre(){
      return nombre;
   }
   public int getDni(){
      return dni;
   }
}
```

```
public class Orden {
    private List<DetalleOrden> items;
    private int id;
    private Date creacion;
    private Date envio;

    public Orden(int id, Date creacion) {
        this.id=id;
        this.creacion=creacion;
        items=new ArrayList<>();
    }
    public void agregarItem(DetalleOrden detalle){
        items.add(detalle);
    }
    public int getNroItems(){
        return items.size();
    }
    public void setEnvio(Date envio){
        this.envio=envio;
    }
}

public class Notebook {
    private String marca;
    private String modelo;
    private String procesador;
    private float tamañoDisco;

    public Notebook(String marca, String modelo, String procesador, float tamañoDisco) {
        this.marca = marca;
        this.modelo = modelo;
        this.procesador = procesador;
        this.tamañoDisco = tamañoDisco;
    }
   public String getMarca() {
        return marca;
    }
    public String getModelo() {
        return modelo;
    }
    public String getProcesador() {
        return procesador;
    }
    public float getTamañoDisco() {
        return tamañoDisco;
    }
}

public class DetalleOrden {
    private Notebook item;
    private float precioUnitario;
    private int cantidad;

    public DetalleOrden(Notebook item, float precioUnitario, int cantidad) {
        this.item = item;
        this.precioUnitario = precioUnitario;
        this.cantidad = cantidad;
    }
```

```
    public int getCantidad() {
        return cantidad;
    }

    public float getPrecioUnitario() {
        return precioUnitario;
    }
    public float getItem() {
        return item;
    }
}
```

Ejercicio 2. Implementar un método a partir de un enunciado

Implemente el método *calcularTotalOrden* en la clase *Orden*. El mismo deberá retornar el precio total de una orden. Para esto deberá tener en cuenta la cantidad solicitada de cada notebook. Por ejemplo, si la orden está compuesta por los siguientes ítems:

- 2 notebooks Abble "Pro" cuyo precio unitario es $20000
- 1 notebook Mamung "Z" cuyo precio unitario es $10000

Entonces, el precio total de la orden será $50000.

```
public float calcularTotalOrden(){
        float precioTotal=0;
        for (Iterator<DetalleOrden> iterator = items.iterator(); iterator.hasNext();) {
            DetalleOrden next = iterator.next();
            precioTotal=precioTotal+next.getCantidad()*next.getPrecioUnitario();
        }
        return precioTotal;
}
```

Ejercicio 3. Interpretación de código

Un desarrollador implementó el siguiente método en la clase Cliente pero no usó nombres representativos. Indique cual de los siguientes seria el nombre más representativo para el método dada su funcionalidad:

A. filtrarOrdenes()
B. obtenerOrdenConMasItems()
C. obtenerOrdenDeMayorPrecio()
D. ordenarOrdenesPorCantidadDeItems()
E. ordenarOrdenesPorPrecio()

La respuesta correcta es: _

```
public void xxx() {
        int n = ordenes.size();
        int k;
        for (int m = n; m >= 0; m--) {
            for (int i = 0; i < n - 1; i++) {
                k = i + 1;
                if (ordenes.get(i).getNroItems() > ordenes.get(k).getNroItems()) {
                    Orden temp = ordenes.set(i, ordenes.get(k));
                    ordenes.set(k,temp);
                }
```

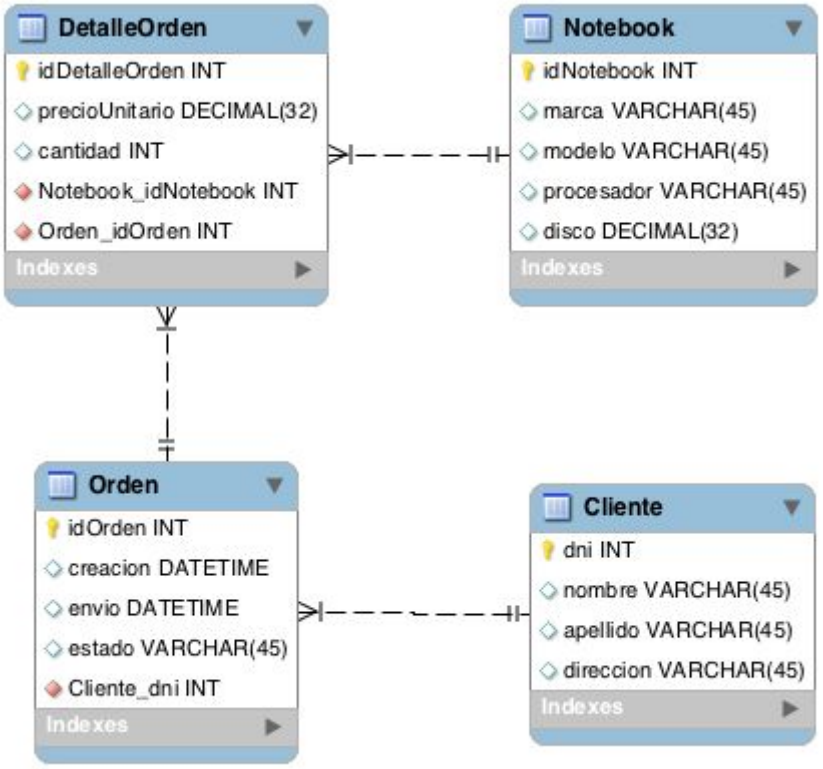
```
        }
    }
}
```

Tenga en cuenta que el método set(i,nuevoElemento) inserta nuevoElemento en la posición i de la lista y retorna el elemento que estaba inicialmente en la posición i.

El método ordena las órdenes de compra de un cliente de menor número de ítems a mayor. Por lo que el nombre más representativo para el método es ordenarOrdenesPorCantidadDeItems()

Ejercicio 4. Interpretación de DER.

A partir del DER, responda Verdadero (V) o Falso (F) a las siguientes afirmaciones (asignar a cada afirmación un valor V o F. Todas las afirmaciones tienen que tener un valor asignado. Puede haber varios V y varios F):



Entre las tablas Notebook y DetalleOrden existe una relación 1 a 1	
Entre las tablas Notebook y DetalleOrden existe una relación 1 a N	
Entre las tablas Notebook y DetalleOrden existe una relación N a N	
La clave primaria de la tabla Orden es idOrden	
La clave foranea de la tabla Orden es idOrden	
La tabla Cliente no tiene clave primaria	
La tabla Cliente no tiene clave foránea	

Entre las tablas Notebook y DetalleOrden existe una relación 1 a 1	**Falso**

Entre las tablas Notebook y DetalleOrden existe una relación 1 a N	**Verdadero**
Entre las tablas Notebook y DetalleOrden existe una relación N a N	**Falso**
La clave primaria de la tabla Orden es idOrden	**Verdadero**
La clave foranea de la tabla Orden es idOrden	**Falso**
La tabla Cliente no tiene clave primaria	**Falso**
La tabla Cliente no tiene clave foránea	**Verdadero**

Ejercicio 5. Consultas SQL

Dado el diagrama de entidad-relación presentado en el ejercicio anterior, escriba una consulta SQL que liste los números de orden (id) de todas las órdenes pertenecientes a clientes con apellido Rodriguez.

SELECT orden.idOrden FROM Orden orden JOIN Cliente cliente ON orden.Cliente_dni=cliente.dni WHERE cliente.apellido="Rodriguez"