**Ejercicio 1**

Dado el siguiente modelo que ya está implementado:



Implemente en Java (solo a nivel del dominio) las siguientes consultas:

1. ¿Cuál es el producto más barato?
2. Dado un cliente y un producto, saber si ese cliente ha comprado o no ese producto.
3. ¿Qué clientes nunca han comprado el producto más barato? La consulta debe retornar una colección de objetos Cliente.

Requerimientos de diseño:

\*Utilizar ***estrictamente*** del diagrama dado.

\*Aplicar Experto

**NOTA:** En el modelo no figuran las operaciones necesarias para el mantenimiento de la información dado que son irrelevantes para el problema planteado.

**Ejercicio 2**

Dado el siguiente modelo que ya está implementado:



Implemente en Java la siguiente consulta:

**Dado un producto x, ¿Cuál es el cliente que (en total) ha gastado más dinero en ese producto?**

Requerimientos de diseño:

\*Utilizar ***estrictamente*** del diagrama dado.

\*Aplicar Experto

No es relevante realizar optimizaciones de performance. Una colección pueda ser recorrida varias veces durante la ejecución de la consulta.

Asuma que no hay más de un cliente que haya gastado igual dinero en un producto.

(en caso de “empate” retornar el ultimo o el primero)

**NOTA:** En el modelo no figuran las operaciones necesarias para el mantenimiento de la información dado que son irrelevantes para el problema planteado.

**Ejercicio 3**

Dado el siguiente modelo que ya está implementado:



Implemente en Java las siguientes consultas:

1) Dado un cliente y un producto, saber cuánto ha gastado ese cliente en ese producto.

2) Dado un monto y un producto saber qué clientes han gastado más de ese monto en ese producto. La consulta debe retornar una colección de objetos Cliente.

**Reglas:**

Requerimientos de diseño:

\*Utilizar ***estrictamente*** del diagrama dado.

\*Aplicar Experto

**NOTA:** En el modelo no figuran las operaciones necesarias para el mantenimiento de la información dado que son irrelevantes para el problema planteado.