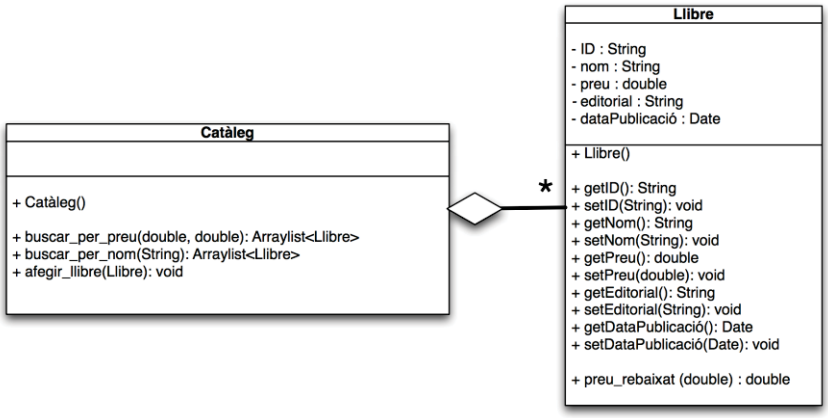
Problema 3. Una empresa dedicada a la venda de llibres realitza la gestió del seu catàleg mitjançant una aplicació basada en el següent diagrama de classes de domini:



Com es pot veure al diagrama, actualment es gestiona l'ID, el nom, el preu, l'editorial i la data de publicació de cada llibre. El mètode "buscar_per_preu" retorna tots els llibres pels quals el seu preu està dins del rang [min, max] definit pels seus paràmetres. El mètode "buscar_per_nom" retorna tots els llibres pels quals el seu nom conté la cadena indicada pel seu paràmetre. El mètode "preu_rebaixat" calcula el preu després d'aplicar el descompte indicat al paràmetre (hem decidit fer-ho en un mètode per si un dia ho hem de canviar per un càlcul més complex). La resta de mètodes són el que es pot deduir del seu nom.

1) Implementeu en Java el mètode "buscar_per_preu" del diagrama de classes. Podeu suposar que l'agregació entre Catàleg i Llibre està implementada amb un *ArrayList*. Recordeu que podeu usar el mètode *int size()* per obtenir la mida d'un *ArrayList*, el mètode *get(n)* per obtenir l'objecte de posició *n* i el mètode *add(E)* per inserir un objecte E.

2) La empresa ha decidit ampliar el seu negoci incorporant altres tipus d'articles al seu catàleg de vendes per a obtenir un millor rendiment. Per això, hem d'adaptar l'aplicació. Modifiqueu el diagrama de classes de la figura anterior per tal que sigui possible gestionar tant llibres com sabates, mantenint totes les funcionalitats que ja tenim pels llibres. Per gestionar les sabates necessitarem les següents funcionalitats:
   a) Gestionar l'ID, el nom, el preu i el material del qual està feta cada sabata.
   b) Buscar sabates per preu i per nom, igual que per llibres.
   c) Calcular el preu rebaixat de la sabata, igual que per llibres.

   Volem gestionar els llibres i les sabates amb catàlegs independents, de manera que quan busquem llibres per preu (o per nom) només obtenim els llibres (i no les sabates) que compleixin la condició. A més, es requereix que la solució permeti aprofitar el màxim possible del codi de la implementació existent quan necessitem incorporar la gestió de nous tipus d'articles en el futur.

3) Implementeu en Java el mètode "buscar_per_preu" (o equivalent/s) del nou diagrama de classes. Quines diferències hi ha amb la implementació original de l'apartat 1?

4) Després d'uns mesos de bon funcionament, l'empresa s'adona que necessita noves funcionalitats específiques per a cada article, com per exemple:
   a) Buscar tots els llibres publicats en un any determinat
   b) Buscar tots els llibres publicats per una editorial determinada
   c) Buscar totes les sabates elaborades amb un material determinat

   Modifiqueu el diagrama de classes que heu fet a l'apartat 2 de manera que es puguin implementar aquestes noves funcionalitats. Novament volem aprofitar tot el codi que sigui possible, com a l'apartat 2.