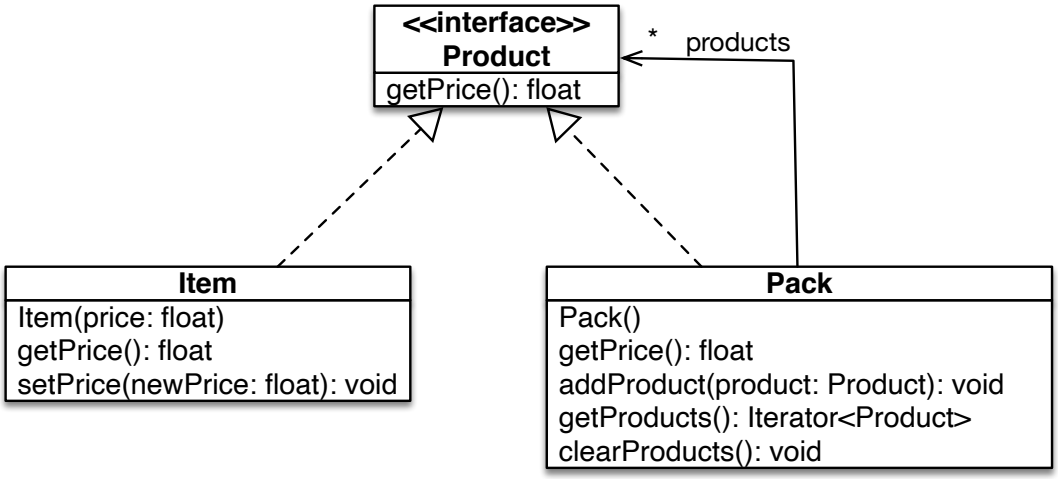
Donat el següent disseny de partida:



La classe `Item`:

- representa un producte individual que, al seu constructor, rep el valor del preu el qual, per simplificar, serà un `float` i que si rep un **preu negatiu o zero** es llença l'excepció no comprovada `IllegalArgumentException`
- té el mètode `getPrice` que retorna el preu del producte
- té el mètode `setPrice` per canviar el preu del producte i si rep un **preu negatiu o zero** es llença l'excepció no comprovada `IllegalArgumentException`

La classe `Pack`

- representa un grup de productes (que poden ser tant `Items` com `Packs`)
- té el mètode `addProduct` per afegir un nou producte al pack
- té el mètode getPrice per calcular el preu del pack que és la suma dels preus dels productes que conté. Per exemple: si els productes tenen preus 10,0 i 20,0, el preu del pack és 30,0
- té el mètode getProducts que retorna un iterador sobre els productes que conté el Pack (només el primer nivell, ja que el Pack, com a composite que és, genera una estructura arborescent).

Com voldrem realitzar vàries operacions sobre els productes hem pensar en aplicar el patró visitor de cara a tenir una infraestructura que ens permeti afegir-les sense tenir que estar constantment redefinint la classe.

a) (1 punt) Definiu la interfície ProductVisitor que permetrà fer-ho.
b) (1 punt) Modifiqueu el disseny original de cara a que es puguin utilitzar aquests visitants dins dels productes. **Mostreu només el codi que heu modificat/afegit**.
c) (4 punts) Implementeu un visitant tal que, donat un producte, augmenti el preu dels ítems dels producte per un determinat factor que serà un float. El factor ha d'estar entre 0 y 1 (0.0 < factor < 1.0) i, per tant, la modificació del preu serà preuNou = preu Antic * (1 + factor). Si el factor no està en el rang donat, es llençarà l'excepció `IllegalArgumentException.` **En aquest apartat podeu suposar que els productes formen una estructura arborescent.**
d) (4 punts) Implementeu un visitant tal que, donat un preuMínim,, que és un float, calculi la llista de Items que tenen un preu per sobre (>=) d'aquest preu mínim. **Aquí haureu de tenir en compte que els productes poden no formar un arbre i que, òbviament, no es vol entrar en cicles.**