# TP8 - Hiérarchie de classes

M2103 - Programmation orientée objets

Département Informatique - IUT de Bordeaux - mars 2016

On considère un système de gestion pour des distributeurs de produits (boissons et de friandises).

## Produit

Le code de la classe `Produit` vous est fourni. Etudiez-le

- Quels services offre-t-elle ?
- Quels attributs/méthodes seront accessibles depuis ses futures sous-classes ?

## Classes Friandise et Boisson

- Une Friandise est un produit qui a un poids.
- Une Boisson est un produit qui a un volume

Ecrivez la déclaration et l'implémentation de ces deux classes, en y ajoutant un accesseur pour connaitre leurs attributs spécifiques (poids et volume).

## Détermination du type

On souhaite être en mesure de déterminer si un Produit désigne ou non un Friandise ou une Boisson.

Il y a plusieurs façons de faire. Essayez les deux :

- ajoutez à `Produit` un prédicat `virtual bool estFriandise() const` redéfini dans les sous-classes.
- définissez une fonction libre `bool estBoisson(Produit *)` qui fait appel au transtypage dynamique.

Avantages et inconvénients des deux ?

## Classe Distributeur

Un `Distributeur` peut stocker un certain nombre (contenance) de Boissons et de Friandises.

### Les actions sur un distributeur

- constructeur paramétré par les contenances
- ajouter un produit `ajouter(Produit *)`. L'opération échouera si la capacité est atteinte.
- faire afficher le contenu du distributeur (compteurs, produits présents, somme en caisse)
- demander un produit en indiquant son identifiant et une certaine somme. `demander(int identifiant, float somme)`. L'opération retournera le produit demandé si il existe et si la somme est suffisante. Si l'opération réussit, il prend l'argent mais ne rend pas la monnaie.
- consulter le contenu de la caisse `float valeurEnCaisse()`

### Représentation

Pour représenter le distributeur, on aura probablement besoin

- de deux variables pour les capacités
- de deux variables pour les nombres de friandises et boissons
- d'une variable pour le montant de la caisse.

et pour le stock de produits, on peut utiliser un tableau associatif (map) qui, à un identifiant de produit, associe le/les produits qui ont cet identifiant.

Par exemple

```
map<int, vector<Produit *>>
```

### Plan de travail

Ecrire les fonctions dans l'ordre, en les testant au fur et à mesure.