

# Exercice P2 – Distributeur de boissons

Le but de cet exercice est de simuler de façon simplifiée un distributeur de boissons. Une version exécutable du programme se trouve dans votre répertoire de classe. Le diagramme de classes se trouve en dernière page.

## Partie Unimozzer

1. Créez le projet **ExerciceP2** dans Unimozzer.
2. Créez la classe `Product` avec les attributs suivants :
  - a. `name` contient le nom du produit.
  - b. `quantity` contient le nombre de produits restants.
  - c. `pricePerItem` contient le prix unitaire du produit.
3. Ajoutez des accesseurs et des manipulateurs pour tous les attributs.
4. Ajoutez un constructeur qui crée un nouveau produit et en initialise les attributs.
5. Ajoutez la méthode `toString` qui retourne une chaîne de caractères de la forme suivante : "<name> - <pricePerItem>€/item - <quantity> remaining"  
Les parties entre < > sont à remplacer par les valeurs réelles des attributs.
6. Ajoutez la classe `VendingMachine` qui gère une liste de produits `alProducts`. Il y a un deuxième attribut `message` qui contient le message actuel du distributeur de boissons. Un troisième attribut `money` contient la somme d'argent insérée dans le distributeur.
7. Ajoutez la méthode `getMessage` qui retourne le message actuel du distributeur.
8. Ajoutez la méthode `cancel` qui fait ressortir l'argent inséré dans le distributeur et adapte le message du distributeur (cf. exécutable).
9. Ajoutez la méthode `addMoney` qui ajoute la somme d'argent indiquée en paramètre à la somme déjà insérée dans le distributeur. Le message du distributeur est adapté (cf. exécutable).
10. Ajoutez la méthode `buy` qui possède deux paramètres, l'index d'un produit et une quantité que le client veut acheter. La méthode vérifie si la quantité indiquée en paramètre n'excède pas la quantité disponible et si l'utilisateur a inséré assez d'argent. Si c'est le cas, le produit peut être acheté en quantité indiquée. Dans tous les cas le message du distributeur est adapté (cf. exécutable).
11. Ajoutez la méthode `searchCheapestProduct` qui cherche et retourne le produit le moins cher.
12. Ajoutez la méthode `searchProductWithName` qui cherche et retourne le premier produit avec le nom indiqué.



## Partie NetBeans

13. Ajoutez les méthodes `isEmpty`, `toArray`, `get`, `add` et `remove` à la classe `VendingMachine`.
14. Créez l'interface graphique de la page suivante. Pensez à renommer les variables des composants graphiques auxquels vous devez accéder à partir du code Java.

15. Ajoutez un attribut `vm` à la classe `MainFrame` pour lui permettre de communiquer avec le modèle et initialisez-le avec un objet de la classe `VendingMachine`.
16. Quand le programme est lancé, seulement le panneau inférieur permettant de remplir le distributeur est visible.
17. Le bouton **Add** permet d'ajouter un nouveau produit au distributeur.
18. Le bouton **Modify** permet de modifier les informations du produit sélectionné dans la liste graphique.
19. Le bouton **Show Cheapest** permet d'afficher les détails du produit le moins cher.
20. Le bouton **Show Quantity and Price** permet de chercher le produit portant le nom indiqué et de remplir les deux autres champs de texte avec les informations correspondantes du produit trouvé.
21. Le bouton **Remove** enlève le produit sélectionné du distributeur.
22. Quand la sélection de la liste graphique change, les valeurs des attributs du produit sélectionné sont utilisées pour remplir les champs de texte.
23. Le bouton **Finish Refill** cache le panneau inférieur (s'il y a au moins un produit dans le distributeur) et affiche le panneau supérieur qui permet d'acheter des produits.
24. Le bouton **Buy Selected** permet d'acheter la quantité indiquée du produit sélectionné dans la liste.
25. Le bouton **Cancel** permet à l'utilisateur de retirer son argent.
26. Les boutons **+1€** et **+5€** permettent d'insérer la somme d'argent indiquée.
27. Le bouton **Refill** fait ressortir l'argent encore inséré dans le distributeur, cache le panneau supérieur et affiche le panneau inférieur qui permet de remplir le distributeur.

### Interface graphique à réaliser

