

MOOC Intro POO C++

Exercices semaine 2

Exercice 5 : Apéritif (niveau 1)

Cet exercice correspond à l'exercice n°49 (pages 119 et 301) de l'ouvrage *C++ par la pratique* (3e édition, PPUR).

Définir la classe `Apero` de sorte que le `main()` suivant :

```
using namespace std;

int main() {
    Apéro bic;
    cout << "Super !" << endl;
    bic.bis();
    cout << "Non merci." << endl;
    return 0;
}
```

affiche le texte suivant :

```
L'heure de l'apéro a sonné !
Super !
Encore une ?
Non merci.
A table !
```

Exercice 6 : Un peu de douceur dans ce monde (niveau 1)

Cet exercice correspond à l'exercice n°50 (pages 119 et 301) de l'ouvrage *C++ par la pratique* (3e édition, PPUR).

Dans un fichier `poeme.cc` définissez la classe `Fleur` de sorte que le programme principal suivant :

```
using namespace std;

int main ()
{
    Fleur f1("Violette", "bleu");
    Fleur f2(f1);
    cout << "dans un cristal ";
    f2.eclore();
    cout << "Donne un poeme un peu fleur bleue" << endl
        << "ne laissant plus ";
    return 0;
}
```

affiche le texte suivant (inspiré d'un poème arabe du 5e siècle de l'hégire) :

```
Violette fraîchement cueillie
Fragile corolle taillée
dans un cristal veine de bleu
Donne un poeme un peu fleur bleue
ne laissant plus qu'un simple souffle...
qu'un simple souffle...
```

Remarque : La solution n'est pas forcément unique.

Exercice 7 : Banque (niveau 3)

Le (très mauvais) programme banque1.cc dont le code est fourni sur Moodle ("code source banque"), contient un programme bancaire qui est modularisé sous forme de fonctions. Transformez-le en programme orienté objet sous le nom de banque2.cc en suivant les étapes suivantes :

- Étudiez le fonctionnement du programme. La banque a 2 clients. Chaque client a un compte privé et un compte d'épargne avec des soldes différents. Le taux d'intérêt d'un compte d'épargne est plus élevé que celui d'un compte privé. Les données de chaque client (nom, ville et soldes) sont affichées avant et après le boucllement des comptes.
- Réfléchissez aux objets que vous aimeriez utiliser dans votre programme et ajoutez les classes correspondantes. Il peut s'agir d'objets de toute nature (client, maison, billet, compte, relation bancaire, etc.). N'oubliez pas que la modularisation n'est pas une science exacte. Chaque programmeur décide des classes qu'il trouve utiles et qui lui semblent correspondre au meilleur modèle de la réalité. C'est souvent l'étape la plus difficile d'un projet de programmation.
- Transférez le code concernant les objets dans les classes. Utilisez le mot-clé `private` pour encapsuler les attributs et les méthodes qui ne seront pas utilisées à l'extérieur de la classe. Chaque méthode doit être courte et sans trop d'instructions détaillées. Les identificateurs (noms des variables, attributs et méthodes) doivent être significatifs.

Exemple d'exécution du programme :

```
Donnees avant le boucllement des comptes :
Client Pedro de Geneve
    Compte courant : 1000 francs
    Compte epargne : 2000 francs
Client Alexandra de Lausanne
    Compte courant : 3000 francs
    Compte epargne : 4000 francs
Donnees apres le boucllement des comptes :
Client Pedro de Geneve
    Compte courant : 1010 francs
    Compte epargne : 2040 francs
Client Alexandra de Lausanne
    Compte courant : 3030 francs
    Compte epargne : 4080 francs
```

Banque avec des client(e)s (Niveau 1)

Vous avez peut-être remarqué que l'affichage du programme précédent ne fait pas de différence entre les clients et les clientes.

Ceci est facile à corriger dans la version orientée objets du programme, par exemple en ajoutant à la classe `Client` (si vous en avez une), une variable d'instance `genre` de type `char` et en testant sa valeur dans la méthode d'affichage (p.ex. 'M', 'F' ou 'A' (ou 'N', pour « non-binaire »).

Modifiez votre programme pour qu'il affiche « Cliente » ou « Client » au lieu de simplement « Client ». Par exemple :

```
Donnees avant le boucllement des comptes :
Client Pedro de Geneve
    Compte courant : 1000 francs
    Compte epargne : 2000 francs
Cliente Alexandra de Lausanne
    Compte courant : 3000 francs
    Compte epargne : 4000 francs
Donnees apres le boucllement des comptes :
Client Pedro de Geneve
    Compte courant : 1010 francs
    Compte epargne : 2040 francs
Cliente Alexandra de Lausanne
    Compte courant : 3030 francs
    Compte epargne : 4080 francs
```

Exercice 8 : Supermarché (niveau 2)

Un supermarché souhaite que vous l'aidiez à afficher le total des achats enregistrés par ses caisses. Il s'agit de compléter le programme `supermarche.cc` fourni sur Moodle ("code source supermarché").

Voici les entités nécessaires pour modéliser le fonctionnement du supermarché :

- les *articles vendus* : caractérisés par leur *nom* (une chaîne de caractères), leur *prix unitaire* (un double) et un booléen indiquant si l'article est *en solde ou pas* (en Suisse, on dit « en action »);
- les *achats* : un achat est caractérisé par l'*article acheté* et la *quantité achetée* de cet article;
- les *caddies* : caractérisés par l'ensemble des achats qu'ils contiennent;
- les *caisses* : chargées de scanner et enregistrer le contenu des caddies; une caisse est caractérisée par un *numéro de caisse* (un entier) et le *montant total des achats* qu'elle a scanné (un double).

Le programme principal fourni a pour but de faire afficher le montant total de chaque caisse au bout d'une journée donnée. **Commencer par l'étudier.**

Coder ensuite les structures de données et les méthodes manquantes. **Ces entités doivent pouvoir être testées avec le programme principal fourni.**

Dans ce fichier, déclarer les classes nécessaires à la modélisation du supermarché, telles que suggérées ci-dessus.

Il est suggéré d'utiliser un vector d'achats pour modéliser le contenu du caddie.

Faire bien attention à l'encapsulation (e.g. les attributs doivent être privés).

Pour les achats, les méthodes à implémenter sont :

- `afficher()` affichant les caractéristiques de l'article (son nom, son prix unitaire, la quantité achetée et le prix de l'achat). De plus, si l'article concerné est en solde, il faudra afficher le texte « (en action) ».

Voici le modèle d'affichage pour `afficher()` :

Petit-lait : 2.5 x 6 = 7.5 F (en action)

où Petit-lait est le nom de l'article, 2.5 son prix unitaire, 6 la quantité achetée, 7.5 le prix de l'achat et « (en action) » l'indication que l'article est en solde. Cette indication ne doit évidemment apparaître que si l'article est en solde.

- Toute autre méthode vous semblant nécessaire.

Pour les caddies :

- `remplir(...)` conforme au programme principal fourni.

Réfléchissez à comment stocker le contenu du caddie (qui sera scanné par la suite).

- Toute autre méthode vous semblant nécessaire.

Pour les caisses :

- `afficher()` qui affiche son numéro et la valeur de son champ montant total selon la forme de l'exemple suivant :

La caisse 1 a encaisse 121.15 F aujourd'hui.

où 1 est le numéro de la caisse et 121.15 le montant total. Vous supposerez que ce montant total est stocké comme attribut (et qu'il est mis à jour par la méthode `scanner(...)`, ci-dessous).

- `scanner(...)` : cette méthode, qui doit être conforme au programme principal fourni, permet à la caisse d'afficher le ticket de caisse correspondant au contenu du caddie. Cette méthode doit aussi **mettre à jour le montant total de la caisse** en y ajoutant le montant des achats du caddie.

L'affichage du ticket de caisse doit se faire selon le modèle ci-dessous et doit utiliser la méthode `afficher` précédemment codée.

Exemple de déroulement :

Une fois le programme complété, l'exécution du programme principal devrait ressembler à ceci :

```
Chou-fleur extra : 3.5 x 2 = 7 F
C++ en trois jours : 48.5 x 1 = 48.5 F
Cookies de grand-mere : 3.2 x 4 = 12.8 F
```

Petit-lait : $2.5 \times 6 = 7.5$ F (en action)

Sardines : $6.5 \times 2 = 13$ F

Total a payer : 88.8 F.

=====

Les malheurs de Sophie : $16.5 \times 1 = 8.25$ F (en action)

Cremeux 100%MG : $5.8 \times 1 = 5.8$ F

Pois surgelés : $4.35 \times 2 = 8.7$ F

Poires Williams : $4.8 \times 2 = 9.6$ F

Total a payer : 32.35 F.

=====

100% Arabica : $6.9 \times 2 = 6.9$ F (en action)

Pain d'epautre : $6.9 \times 1 = 6.9$ F

Cremeux 100%MG : $5.8 \times 2 = 11.6$ F

Total a payer : 25.4 F.

=====

Resultats du jour :

Caisse 1 : 121.15 F

Caisse 2 : 25.4 F