premiers exercices sur les classes

Exercice 2.1.1 *utilisation d'une classe*

Voici le texte d'une classe représentant de façon sommaire un compte bancaire et les opérations bancaires courantes.

```
class Compte{
    int solde = 0;
    void deposer(int montant){
        solde = solde + montant;
    }
    void retirer(int montant){
        solde = solde -montant;
    }
    void virerVers(int montant, Compte destination){
        this.retirer(montant);
        destination.deposer(montant);
    }
    void afficher(){
        Terminal.ecrireString("solde:␣"+ solde);
    }
}
```

Question 1

Comment fonctionne la méthode virerVers ? Combien de comptes fait-elle intervenir ?

Question 2

Créez deux comptes que vous affecterez à deux variables. Ecrivez le code correspondant aux opérations suivantes :

- dépot de 500 euros sur le premier compte.
- dépot de 1000 euros sur le second compte.
- retrait de 10 euros sur le second compte.
- virement de 75 euros du premier compte vers le second.
- affichage des soldes des deux comptes.

Vous mettrez le code java correspondant à cette question dans la méthode `main` d'une nouvelle classe appelée `TesteCompte`. Vous compilerez et testerez ce programme.

Question 3

Créez un tableau de dix comptes. Pour cela, notez bien qu'il faut d'abord créer le tableau puis créer successivement les dix comptes à mettre dans les dix cases de ce tableau.

Dans chaque case, faites un dépot de 200 euros plus une somme égale à 100 fois l'indice du compte dans le tableau.

Ensuite, vous ferez un virement de 20 euros de chaque compte vers chacun des comptes qui le suivent dans le tableau (par exemple, du compte d'indice 5, il faut faire des virements vers les comptes d'indice 6, 7, 8 et 9).

Enfin, vous afficherez les soldes de tous les comptes.

Ici encore, vous testerez et compilerez le code proposé.

Exercice 2.1.2 *constructeurs*

Cet exercice reprend la classe `Compte` de l'exercice précédent.

Question 1

Complétez la classe `Compte` avec une information supplémentaire : le nom du titulaire du compte (type `String`). Vous modifierez la méthode d'affichage pour qu'elle affiche cette information.

Question 2

Créez un constructeur pour la classe `Compte`. Ce constructeur doit prendre en paramètre le nom du titulaire du compte.

Donnez le code de création d'un compte qui appelle ce constructeur.

Question 3

Faut-il prévoir des méthodes permettant de changer le nom du titulaire du compte ?

Exercice 2.1.3 *méthodes statiques ou non*

Parmi les méthodes de la classe suivante, lesquelles peuvent être statiques et lesquelles ne peuvent en aucun cas être statiques ?

```
class Exo12_3{
    int x, y;
    String nom;
    void afficher(){
        Terminal.ecrireString(nom + "␣" + x + "␣␣" + y);
    }
    void ajouter(Exo12_3 obj){
        x = x + obj.x;
        y = y + obj.y;
        nom = nom + obj.nom;
    }
    Exo12_3 nouveau(int n){
        Exo12_3 res = new Exo12_3();
```

```
        res.x = n;
        res.y = n*2;
        res.nom = "Auto_"+n;
        return res;
    }
    boolean plusGrand(Exo12_3 obj){
        if (obj.x == x){
            return y>obj.y;
        }else{
            return x>obj.x;
        }
    }
    boolean compare(Exo12_3 obj1, Exo12_3 obj2){
        if (obj1.x == obj2.x){
            return obj1.y>obj2.y;
        }else{
            return obj1.x>obj2.x;
        }
    }
}
```

Exercice 2.1.4 *égalité d'objets*

```
class Exo12_4{
    public static void main(String[] argv){
        Compteur c1, c2, c3;
        c1 = new Compteur(0);
        c1.incremente();
        c2 = new Compteur(1);
        c3 = c1;
        if (c1 == c3){
            Terminal.ecrireStringln("c1 et c3 sont égaux");
        }else{
            Terminal.ecrireStringln("c1 et c3 ne sont pas égaux");
        }
        if (c1.getValeur() == c2.getValeur()){
            Terminal.ecrireStringln("c1 et c2 ont même valeur");
        }else{
            Terminal.ecrireStringln("c1 et c2 n'ont pas la même valeur");
        }
        if (c1 == c2){
            Terminal.ecrireStringln("c1 et c2 sont égaux");
        }else{
            Terminal.ecrireStringln("c1 et c2 ne sont pas égaux");
        }
        if (c1.getValeur() == c1.incremente().getValeur()){
            Terminal.ecrireStringln("c1 et c1 incremente ont même valeur");
        }else{
            Terminal.ecrireStringln("c1 et c1 incremente n'ont pas la même valeur");
        }
        if (c1 == c1.incremente()){
```

```
            Terminal.ecrireStringln("c1 et c1 incremente sont égaux");
        }else{
            Terminal.ecrireStringln("c1 et c1 incremente ne sont pas égaux");
        }
    }
}
class Compteur{
    int x;
    Compteur(int n){
        x=n;
    }
    Compteur incremente(){
        x++;
        return this;
    }
    int getValeur(){
        return x;
    }
}
```

Essayez de prédire le résultat de l'exécution de ce programme. Testez le programme. Que peut-on en déduire sur la notion d'égalité d'objets en java ?

Notez que l'expression `c1.incremente().getValeur()` comprend deux appels de méthodes successifs. D'abord la méthode `incremente` est appelée sur `c1`. Cette méthode renvoie un objet de type `Compte` qui est celui sur lequel la méthode est appelée (`this`). Sur cet objet, la méthode `getValeur` est appelée. Cela revient à appeler successivement les deux méthodes sur le même objet.

Exercice 2.1.5 *conception*

Cet exercice a pour but de réfléchir sur la conception d'un programme, sa structuration en classes. Il ne s'agit pas pour le moment de réaliser ce programme, mais juste de concevoir son architecture.

On fait des cocktails avec différents liquides (alcools, sodas, jus de fruits). On a un bar avec des bouteilles qui peuvent être pleines ou à moitié vides. On a des shakers qui ont une contenance donnée. Il y a des recettes de cocktails qui indiquent seulement les proportions. Ces recettes peuvent s'appliquer à des quantités plus ou moins grandes selon les besoins du moment.

Les cocktails se font en déversant une partie du contenu des bouteilles dans des shakers. Après, il faut secouer. Les shakers sont ensuite vidés (dans les verres, mais on ne tiendra pas compte des verres dans cette application). Il faut les laver après usage.

Question : quelles classes faut-il créer ? Quelles informations faut-il dans chaque classe ? Quelles méthodes faut-il écrire, et dans quelle classe les mettre ? Pour chaque méthode, précisez le type des paramètres et de la valeur de retour.