Imbrication des constructeurs tableau et objet (TD 8)

Exercices table : 3
Rappel : Respectez les consignes de la page *Fonctionnement* et celles données dans les feuilles précédentes.

1. Placez-vous dans le répertoire adéquat demandé dans la page *Environnement*.
2. **Familiarisation** Tapez et exécutez le programme suivant de la feuille d'exemples du cours, en respectant les consignes de la page *Fonctionnement* : *Ex1bis.java*.
3. On veut reprogrammer avec la totalité du matériel vu jusqu'à présent l'ensemble des exercices du TD 6 (objets et fonctions). Sauf que dans cet exercice, *il ets interdit d'utiliser des fonctions*. A partir du corrigé machine du TD 6 disponible sur la page, faites dans l'ordre les questions ci-dessous pour le transformer progressivement. Après chaque question, compilez, exécutez et testez votre programme avant de commencer la question suivante.

   On redonne ci-dessous l'énoncé complet.

   On considère des comptes bancaires simples, avec un numéro, un solde et un salaire (des entiers). Le "salaire" est la rentrée mensuelle du détenteur du compte. Chaque compte a aussi une autorisation de découvert, dont le montant est un pourcentage du salaire. Le taux (pourcentage) est le même pour tous les comptes.

   On considère des clients, avec un nom, une ville, et un compte (un objet Compte). Saisir deux clients. Afficher toutes les informations de l'application.

   Pour simplifier, deux clients ne peuvent avoir le même compte. On devra pouvoir accéder aux comptes sans passer par les clients. On devra aussi pouvoir accéder aux clients sans passer par les comptes. On a besoin d'accéder, à partir d'un client, à son compte, mais jamais au client à partir de son compte. Il ne peut pas y avoir de compte sans client.

   Saisir le taux au clavier. Saisir un nombre quelconque de clients au clavier avec leur compte (dans l'ordre à chaque fois : nom, ville, numéro, solde, salaire). Afficher toutes les informations de l'application : le taux, les comptes, les clients avec leur compte. Afficher le montant de l'autorisation de découvert maximale, *mais sans passer par le tableau des comptes*.

   (a) Créez la classe Compte.
   (b) Créez la classe Client.
   (c) Choix des structures de stockage.
   (d) Saisie. Vous lirez toutes les informations sur la ligne de commande comme lors de l'interogation. Dans cette question les informations concernant les clients seront présentes sur la ligne de commande mais vous les sauterez.
   (e) Affichage comptes.
   (f) Saisie clients.
   (g) Affichage clients.
   (h) Autorisation découvert.

4. Reprogrammez l'exercice précédent avec des fonctions. (Vous utiliserez bien sûr les exercices précédents de ce TD et des précédents.) Dans toute la suite du module, on fera maintenant toujours directement des fonctions, sauf mention contraire.

5. On reprend l'exercice précédent. Un client peut maintenant avoir plusieurs comptes, au maximum 10, ou aucun. Pour chaque client, le nombre de comptes est fixé à la saisie et ne variera plus ensuite. Vous choisirez les conventions de saisie. Vous devrez bien sûr reprogrammer la saisie. *Il est interdit d'utiliser des fonctions.*

6. *Déroulement mémoire. Travail personnel.*
   On considère l'exécution du programme suivant (il compile sans erreur).

   (a) Donner la configuration de la mémoire immédiatement après l'exécution de la ligne contenant le commentaire : `// ici`. Vous utiliserez un croquis détaillé et l'algorithme vu en cours et TD. En particulier, on allouera obligatoirement les cases mémoire dans l'ordre croissant à partir de 101. Comme en cours et en TD, si une case contient successivement plusieurs valeurs, on les écrira de gauche à droite dans la case en les barrant d'un seul trait au fur et à mesure. De même, on ne réutilisera pas les zones de fonctions.

   (b) Donner le dernier état de la mémoire avant l'arrêt du programme.

   (c) Dire ce qu'affiche ce programme.

```
class Machin {
    int valeur;
}
class Bidule {
    static int d;
    int valeur;
    Machin obj;

    static int f (Bidule b, Bidule c) {
        d++;
        b.obj = new Machin();
        b.valeur = d;
        c = b;
        b.obj.valeur = c.valeur;
        c.valeur++;
        return b.valeur;
    }
}
class Memoire {
    public static void main (String[] args ) {
        Bidule[] t = new Bidule[2];
        t[1] = t[0];
        t[0] = new Bidule();
        int n = 0;
        int m = Bidule.f(t[n], t[n+1]);
        System.out.println(m+" "+n+" "+t[n].valeur+" "+
                           t[n].obj.valeur+" "+Bidule.d); // ici
        m = Bidule.f(t[n+1], t[n]);
        System.out.println(m+" "+n+" "+t[n].valeur+" "+
                           t[n].obj.valeur+" "+Bidule.d);
    }
}
```