Tableaux et Itérations

1. Les tableaux

En JAVA, les tableaux fournissent des collections ordonnées d'éléments. Les composants d'un tableau peuvent être des types de base ou références sur des objets, y compris des références sur d'autres tableaux (les tableaux sont effectiement des objets en JAVA).

a) déclaration de tableaux unidimensionnels

```
typeDesElements[]       nomDuTableau;
```

`typeDesElements` est l'un des types de base du langage JAVA (char, int, float, double...) ou bien un nom de classe.
`nomDuTableau` est l'identificateur choisi pour désigner le tableau.

(Si on préfère, on peut placer les crochets après l'identificateur, `typeDesElements nomDuTableau[];`, néanmoins le premier style est préférable car il place la déclaration de type entièrement en un seul endroit).

Attention : la taille d'un tableau n'est pas spécifiée à la déclaration. Le nombre d'éléments d'un tableau est déterminé quand l'objet est créé en utilisant `new`. Cette taille fixée à la création est fixe, elle ne peut plus être modifiée par la suite.

ex 1:

```
int [] tab = new int[10];     // déclaration et création d'un tableau de 10 entiers
```

attention : lorsque la taille d'un tableau est définie par une constante, on utilisera de préférence une constante nommée.

ex 2:

```
class GroupeTD {
   static final int nbElevesMax =30;
   Eleve[] lesEleves = new Eleves[nbElevesMax];

   for (i = 0; i < nbElevesMax; i++)
               ....

}
```

au lieu de

```
class GroupeTD {

  Eleve[] lesEleves = new Eleves[30];

  for (i = 0; i < 30; i++)
                ....

}
```

b) référence à un élément d'un tableau unidimensionnel

attention : en JAVA comme en C les éléments d'un tableau sont indicés de `0` à `length-1`(`length` est une variable d'instance `public` des objets tableau, pour connaître la taille d'un objet tableau referencé par une variable `t` on utilsera l'expression `t.length`). L'accès à un élément d'un tableau s'effectue à l'aide d'une expression de la forme :

```
expression
      ==>   nomDuTableau[expression1]
```

$expression_1$ doit délivrer une valeur entière (`>= 0` et `<length`)
`expression` délivre l'élément d'indice $expression_1$ du tableau `nomDuTableau`
On peut rencontrer une telle expression aussi bien en partie gauche qu'en partie droite d'affectation.

ex 3 :

```
i = tab1[9];          /* affecte à i la valeur du 10 élément de tab1 */
tab2[i+j] = x / 3.0; /* affecte à l'élément de tab2 dont l'indice est la valeur de
                        l'expression i+j la valeur de l'expression x / 3.0 */
```

attention : quand on déclare un tableau d'un type d'objets, on déclare un tableau de variables de ce type. Dans l'exemple 2, après le `new` initial, `lesEleves` a une référence sur un tableau de 30 variables qui sont toutes initialisées à `null`. Les objets doivent être ensuite créés explicitement.

ex 4 :
```
class GroupeTD {

        static final int nbElevesMax =30;
        Eleve[] lesEleves = new Eleves[nbElevesMax];

        ...


        for (i = 0; i < lesEleves.length; i++)
        {
               ...
               // récupération dans une varaible nom du nom de l'élève i
               // et dans une variable age de l'age de l'élève i
               //
               ...
               lesEleves[i] = new Eleves(nom,age);  // creation du ième élève
        }

}
```

c) tableaux multidimensionnels

En JAVA, un tableau mutlidimensionnel est considéré comme un tableau dont les éléments sont eux mêmes des tableaux. Un tableau à deux dimensions se déclarera de la manière suivante :

```
typeDesElements[][]     nomduTableau;
```

Comme pour les tableaux à une seule dimension, la longueur de chacune des dimensions du tableau n'est pas spécifiée à la déclaration. Celle ci n'est indiqué que lors de la création, et encore, ceci n'est obligatoire que pour la première dimension. Les autres dimensions peuvent n'être spécifiées que lors des la création effective des tableaux correspondants.

ex 5:

```
double float[][] matrice = new float[4][4];
```

est équivalent à

```
double float[][] matrice = new float[4][];

for (int i=0; i < 4; i++)
      matrice[i] = new float[4];
```

La première notation est un raccourci qui permet d'éviter un ensemble imbriqué d'instructions `new`. Un avantage des tableaux de tableaux est que chaque tableau imbriqué peut avoir une taille différente.

L'accès aux éléments d'un tableau à deux dimensions se fera par une expression de la forme

expression
==> nomDuTableau[$expression_1$][$expression_2$]

$expression_1$ doit délivrer une valeur entière (>= 0 et <nomDuTableau.length) et $expression_2$ doit délivrer une valeur entière (>= 0 et <nomDuTableau[$expression_1$].length).

Ces définitions se généralisent aux tableaux de dimension supérieure à 2.

Exercices
Itérations et tableaux

1) Utilisation d'instructions itératives 'for'

a) Etant donné un entier positif n, calculer la somme des entiers pairs compris entre 1 et n.

b) Etant donné un entier positif a, calculer les 15 premiers termes de la progression géométrique de raison a et de premier terme a.

c) Les premiers nombres de Fibonacci sont : F0 = 1, F1 = 1, F2 = 2, F3 = 3, F4= 5, F5 = 8, ... Chaque nombre est obtenu en faisant la somme des deux nombres précédents, c'est-à-dire :
Fn = Fn-1 + Fn-2 pour n ≥ 2. Calculer les 20 premiers nombres de Fibonacci.

2) Utilisation d'itérations emboîtées (Jones)

a) Ecrire un programme qui lit deux entiers positifs H et L et qui imprime un des motifs suivants :

```
Exemple :   *     *     *     *     *     * * * * *
   H = 4    *     *     *     *     *      * * * *
   L = 5    *     *     *     *     *     * * * * *
            *     *     *     *     *      * * * *
```

b) Ecrire un programme qui lit un entier positif T et qui imprime un des motifs suivants :

```
            *     *     *     *     *                *
Exemple :   *     *     *     *                    * *
    T = 5   *     *     *                        * * *
            *     *                            * * * *
            *                                * * * * *
```

2) Utilisation de tableaux de dimension 1

a) Soit T un tableau de 50 nombres réels, supposé initialisé.
Calculer le maximum de ces nombres, ainsi que le premier indice où apparaît ce maximum ; que faut-il changer pour obtenir le dernier indice ?

b) Soit un tableau T de 100 entiers, supposé initialisé. Déterminer si la suite d'entiers représentée dans ce tableau est strictement croissante.

4) Utilisation de tableaux de dimension 2

a) (Prather)
Soit M un tableau [10],[10] de réels, supposé initialisé. Calculer :
1°) le maximum des éléments de la ième ligne,
2°) le minimum des éléments de la jème colonne,
3°) la somme des éléments de la diagonale,
4°) la somme de tous les éléments.
Reprendre l'exercice en groupant les calculs 1°), 2°), 3°) en une seule itération.

b)
1°) Initialiser un tableau M [5][5] d'entiers (compris entre -10 et +10) en utilisant le générateur aléatoire `random` de la classe `Math`.
2°) Même question, mais on veut obtenir un tableau M symétrique.
3°) Imprimer le tableau M (sous forme d'un tableau à 5 lignes et 5 colonnes).

5) TP

modifier le programme du TP5 (gestion des batiments) de manière à ce que lorsque l'utilisateur quitte l'application les batiments saisis soient affichés dans l'ordre des impots décroissant..

indications :

- déclarer un tableau dans la classe InterfaceTP5,
- à chaque création d'un nouveau batiment (Batiment, Usine, ou Villa) le stocker dans le tableau,
- lorsque le bouton quit est activé, effectuer le tri et afficher chacun des batiments.

Ne pas oublier de prévoir le cas où le tableau est plein.

Travail supplémentaire (pour ceux qui ont fini la question précédente) :

Au lieu d'un tableau dont la taille est fixe, utiliser un objet de la classe Vector (voir documentation en ligne sous netscape) qui fourni un tableau redimensionnable d'objets.