Université de Nice – Sophia Antipolis — Deug MASS MIAS SM 1re année
UEF1 : Informatique & Programmation — 2000–2001

TP N° 10

Premiers tableaux

Ça va mieux en le révisant ... Un tableau `t` permet de collectionner des valeurs en les numérotant, à condition qu'elles aient toutes le même type[1]. Ce type, qu'il s'agisse d'un type primitif ou bien d'un type d'objet, est précisé lors de la *déclaration* du tableau. La taille du tableau est fixée une fois pour toute lors de sa *création* ; cette longueur est consignée dans la variable `t.length`. Chaque case est repérée par `t[`*indice*`]`, les indices prenant leurs valeurs de 0 à `t.length`-1.
Un tableau permet un traitement uniforme des données qui le composent ; il ne faut pas oublier d'initialiser chacune de ses cases avant d'utiliser leur valeur. Le parcours entier d'un tableau est réalisé grâce à une simple boucle `for`. Un parcours susceptible d'être interrompu au beau milieu du tableau se fait généralement au moyen d'une boucle `while` ou `do...while`.

1 Création de tableaux

Exercice 1) 1. Ecrivez une classe qui commence par demander un nombre entier n compris entre 10 et 20 à l'utilisateur.
2. Construisez le tableau des carrés des entiers de $[0, n]$. Faites le afficher sur une même ligne, les éléments séparés par des espaces.
3. Faites calculer et afficher par cette classe la somme des carrés des entiers de l'intervalle [0,10].

Exercice 2) Nombres de Fibonacci La suite de Fibonacci est une suite d'entiers ainsi définis :

$$f_0 = 0$$
$$f_1 = 1$$
$$f_n = f_{n-1} + f_{n-2}$$

1. Trouvez *à la main* les six premiers termes de cette suite.
2. Implantez une méthode pour remplir de proche en proche un tableau afin qu'il contienne les 20 premiers nombres de Fibonacci, le i^{e} nombre étant stocké dans la case d'indice $i - 1$.
3. Procédez à l'édition soignée des 20 premiers éléments de cette suite, à raison de 5 par ligne.
4. Renouvelez l'exercice précédent de façon à ce que l'ordinateur stoppe l'affichage du tableau au premier nombre affiché qui dépasse cent.
5. Faites calculer puis afficher le nombre d'éléments pairs de ce tableau.
6. Ecrivez une fonction `fibTab`(...) prenant en paramètre un entier n supérieur ou égal à 2 et renvoyant comme résultat un tableau d'entiers contenant exactement les nombres de Fibonacci $f_0, ..., f_n$.
7. En utilisant la fonction `fibTab`(...), écrire une fonction `fib`(...) prenant en argument un entier positif ou nul et qui renvoie le nombre de Fibonacci f_n.

1. Pour l'instant, nous ne parlons que des tableaux à une seule dimension.

2 Modéliser avec des tableaux

Exercice 3) On décide d'implanter une classe correspondant aux objets géométriques que sont les polygones. A terme, on pourra élaborer une applet capable d'afficher des polygones de formes variées. Un polygone sera stocké sous forme de tableau de points. Ainsi, un triangle sera représenté par un tableau de trois points, un losange par un tableau de quatre points...

1. Commençons à écrire la classe `Polygone`. Que prendrez-vous comme seule variable privée des instances de la classe `Polygone`?
2. L'unique constructeur de cette classe `Polygone` consiste en la déclaration d'un tableau de `n` cases à partir de l'entier n passé en paramètres au constructeur. Ecrivez ce constructeur.
3. Ecrivez une méthode qui place les coordonnées du i^{e} point du polygone dans le tableau qui le représente. Voici l'entête de cette fonction :
   `public void mettrePointIci(Point p, int i)`
4. Ajoutez à votre classe une méthode d'entête :
   `public String toString`(...)
   permettant de renvoyer sous forme d'une chaîne de caractères des plus sommaires les caractéristiques d'un polygone donné.
5. Ecrivez une classe de test `TP10Ex3` qui crée le triangle `t` défini par les trois point $\binom{0}{0}$, $\binom{1}{-1}$ et $\binom{2}{1}$. Créez ensuite un pentagone choisi entièrement aléatoirement dans le cadran délimité par l'origine et le point $\binom{10}{10}$.
6. Faites afficher les informations relatives à ces deux instances de la classe `polygone` par le simple appel à :
   `System.out.println("Descriptifs : triangle " + t + ", pentagone : " + p);`

3 Approfondissement

Exercice 4) Munissez la classe `Polygone` d'une méthode `dessine`(...). Elle prendra comme seul argument un objet de la classe `Graphics`. Elle tracera l'une après l'autre chaque ligne du polygone auquel on l'applique.
Affichez le triangle précédent, un pentagone aléatoire ... et même, laissez libre cours à votre imagination !