LES TABLEAUX ET STRUCTURES

1 Notion de tableau

Soit un entier n positif.
Supposons qu'on veuille saisir n valeurs réelles afin de calculer leur moyenne, ou de trouver leur minimum, ou de les afficher par ordre croissant.
Pour des valeurs petites de n, on peut déclarer n variables réelles pour résoudre le problème. Mais si n est assez grand, on se rend compte que cela devient impropre, fastidieux, voire impossible.
Il faudrait, dans ce cas, utiliser une variable permettant de représenter les n valeurs. Le type de données de cette variable serait le type tableau.

Définition :
Un tableau est une collection séquentielle d'éléments de même type, où chaque élément peut être identifié par sa position dans la collection. Cette position est appelée indice et doit être de type scalaire.
2

Déclaration :
Pour déclarer un tableau, il faut donner :
- son nom (identificateur de la variable)
- ses bornes : la borne inférieure correspondant à l'indice minimal et la borne supérieure correspondant à l'indice maximal.
- le type des éléments le composant.

Syntaxe :
type
nom = tableau[<indice minimum> .. <indice maximum>] de <type des composants>
ou
Variable
nom : tableau[<indice minimum> .. <indice maximum>] de <type des composants>

Exemple :
variable t : tableau[1..10] de réels

Schématiquement, on va représenter la variable t comme suit:

1	2	3	4	5	6	7	8	9	10
8.4	3.5	12	20	10	13.34	50	100	30.1	60.9

Dans la mémoire centrale, les éléments d'un tableau sont stockés de façon linéaire, dans des zones contiguës.
Le tableau ci-dessus est de dimension 1, nous verrons un peu plus loin que l'on peut représenter des tableaux à 2 dimensions, voire même plus.
L'élément n° I sera représenté par l'expression t[I].
Dans notre exemple, t[I] peut être traité comme une variable réelle. On dit que le tableau t est de taille 10.

2 Création d'un tableau

La création d'un tableau consiste au remplissage des cases le composant. Cela peut se faire par saisie, ou par affectation.
Par exemple, pour remplir le tableau t précédent, on peut faire :
t[1] := 8.4 ; t[2] := 3.5 ; … t[10] := 60.9 ;
Si on devait saisir les valeurs, il faudrait écrire :

```
Pour i := 1 à 10 faire
Ecrire("saisir un élément:")
lire(t[i] );
Fpour
```

3 Affichage d'un tableau

Afficher un tableau revient à afficher les différents éléments qui le composent. Pour cela, on le parcourt (généralement à l'aide d'une boucle avec compteur) et on affiche les éléments un à un.

```
Pour i := 1 à 10 faire
Ecrire(t[i])
Fpour
```

4 Traitement d'un tableau

Après avoir créé un tableau, on peut y effectuer plusieurs opérations comme le calcul de la somme ou de la moyenne des éléments, la recherche du plus petit ou du grand élément du tableau, le test d'appartenance d'un objet au tableau, …
Pour la suite, on considère un tableau d'entiers t déclaré comme suit :

```
Variable t : Tableau[1 .. n]: Entier ;
```

4.1 Somme des éléments d'un tableau

On effectue la somme des éléments du tableau t, le résultat est dans la variable S :

```
S ←-0 ;
Pour i<--1 à n faire
        S <--S + t[i];
```

Ecrire(“la somme des éléments de t est:”, S);

4.2 Minimum d’un tableau

On cherche le plus petit élément du tableau t, le résultat est dans la variable min :

```
   min ←- t[1] ;
Pour i<--1 à n faire
            Si t[i] < min  Alors
             min ←-t[i];
Fpour
   Ecrire(“Le minimum des éléments de t est:”, min) ;
```

4.3 Test d’appartenance

On cherche si l’entier x appartient à t, le résultat est mis dans la variable booléenne appartient :

```
appartient ←-false
Pour i<--1 à n faire
         si t[i]=x Alors
                  appartient := VRAI;
SI appartient Alors
         Ecrire(‘x appartient à t’)
Sinon
  Ecrire(‘x n’’appartient pas à t’);
```

On remarque que l’on peut arrêter les itérations (la recherche) si l’on rencontre l’élément x dans t. Pour cela, il faut utiliser une boucle Tant Que ou Répéter:

```
i ←- 1;
Tant que (t[i]<>x) et  (i<=n)  Faire
      i←-i+1;
Si i>n Alors
      Ecrire(“x n’’appartient pas à “’)
Sinon
      Ecrire(‘x appartient à t’);
```

```
i ←--0 ;
Répéter
         i←-i+1;
Jusqu’à (t[i]=x) ou (i>n);
Si i>n Alors
      Ecrire (‘x n’’appartient pas à t’)
Sinon
      Ecrire(‘x appartient à t’);
```

Dans les deux cas, si x appartient à t, la valeur de i est l’indice de la case qui le contient.

5 Tableaux de caractères

Notion de chaîne de caractères

Une chaîne de caractères est soit une chaîne vide, soit un caractère suivi d'une chaîne de caractères ; en un mot c'est une collection de caractères.
Exemples : "Bonjour", "L'UVS se situe à la VDN", "A", "2017"
La plupart des langages de programmation, Pascal notamment, disposent d'un type chaîne de caractères et d'un ensemble de fonctions prédéfinies permettant de traiter les chaînes de caractères. Ces fonctions permettent de calculer la longueur d'une chaîne, de concaténer deux chaînes, d'extraire une sous-chaîne, de comparer deux chaînes, etc.
Il faut noter qu'une chaîne de caractères peut être traitée comme un tableau de caractères.
En Algorithme, une variable de type chaîne est une séquence de caractères de longueur variable au cours de l'exécution et une taille prédéfinie entre 1 et 255.

Exemple :
variable s : chaine; (* 255 caractères sont alors réservés *)
s10 : chaîne[10] ; (* seuls 10 caractères sont réservés *)

On peut appliquer les opérateurs suivants sur des variables de type Chaîne :
+, =, <>, <=, >=, <, >.
Les fonctions prédéfinies les plus usuelles sont :

- longueur(s :Chaîne) : entier;

retourne la longueur courante de s.

- copier(s : Chaine ; p : entier ; l :entier): Chaine;

renvoie une chaîne constituée de l caractères à partir de la position p.

- concat(s1,s2 : Chaine): Chaine ;

renvoie la chaîne résultant de la concaténation de s1 et de s2.

- pos (ssch : Chaîne; ch : Chaine): Entier;

renvoie la position du premier caractère de la sous-chaîne ssch dans la chaîne ch ; si la sous-chaîne ne se trouve pas dans la chaîne, elle renvoie 0.

Tableau à deux dimensions:

L'informatique nous offre la possibilité de déclarer des tableaux dans lesquels les valeurs ne sont pas repérées par une seule, mais par deux coordonnées.

Un tel tableau se déclare ainsi :

```
Tableau Cases[L,C] : Entier
```

Cela veut dire : réserve moi un espace de mémoire pour L x C entiers, et quand j'aurai besoin de l'une de ces valeurs, je les repèrerai par deux indices.

Cases[i,j] représente la case à la ligne i, colonne j.

→Remarque essentielle :

Il n'y a aucune différence qualitative entre un tableau à deux dimensions (i, j) et un tableau à une dimension (i * j).

Cases[i,j] est équivalent à **T[(C-1)*i+j]** , où T est un tableau à une dimension de L*C entiers.

Le type structure permet de regrouper de éléments de types différents, contrairement au constructeur tableau qui regroupe plusieurs éléments du même type. On définit donc le type

b. Les Structures

structure = enregistrement comme la donnée de *n* variables : $ch_1, ch_2, \ldots, ch_n$ qui seront appelées les champs de la structure et qui seront de types respectifs : $type_1, type_2, \ldots, type_n$.

On adoptera le code suivant pour représenter le type structure.

Type Structure= enregistrement

ch_1 : $type_1$

ch_2 : $type_2$

. .

ch_n : $type_n$

Fin Structure

Remarque : *Dans une structure, tous les noms de champs doivent être distincts. Par contre rien n'empêche d'avoir 2 structures avec des noms de champs en commun : l'ambiguïté sera levée par la présence du nom de la structure concernée.*

L'accès ou l'utilisation d'un champ peuvent se faire par exemple par $T.\ ch_j$ où *T* est une variable de type *Structure*.

Exemple

Structure = point

abs : réel

ord : réel

Fin structure

Variable

P : Point

Tests

Un tableau est :

1. Une suite de chaînes de caractère
2. Une collection séquentielle d'éléments de même type
3. un ensemble de données

Dans la mémoire centrale, les éléments d'un tableau sont stockés :

1. de façon linéaire
2. de façon aléatoire

Les éléments d'un tableau sont affichés :

1. Un à un
2. Deux à deux
3. En vrac

Après avoir créé un tableau on peut y effectuer

1. Une seule opération
2. plusieurs opérations
3. Aucune opération

Quel le résultat de cet algorithme?

```
S ←-0 ;
Pouri<--1 à n faire
        S <--S + t[i];
Ecrire("le résultat est:", S);
```

1. Calculer la somme des éléments du tableau
2. Calculer le produit des éléments du tableau
3. Trier les éléments du tableau

Une chaîne de caractère :

1. c'est une collection de mot
2. c'est une collection de caractères
3. c'est une collection de caractères spéciaux

Une variable de type chaîne a une taille prédéfinie entre :

1. 0 à 255
2. 0 à 128
3. 1 à 255

La fonction prédéfinie longueur (s) : entier; retourne :

1. la position de s
2. la longueur courante de s
3. la position de s