Chapitre 03 : Les Tableaux

I. Introduction

Imaginons que dans un programme, nous avons besoin d'un grand nombre de variables, il devient difficile de donner un nom à chaque variable.

Exemple :

Ecrire un algorithme permettant de saisir cinq notes et de les afficher après avoir multiplié toutes les notes par trois.

Solution :

```
Algorithme  Note
Variables   N1, N2, N3, N4, N5 : Réel
Début
   Ecrire ("Entrer la valeur de la 1er note")
   Lire (N1)
   Ecrire ("Entrer la valeur de la 2ème note")
   Lire (N2)
   Ecrire ("Entrer la valeur de la 3ème note")
   Lire (N3)
   Ecrire ("Entrer la valeur de la 4ème note")
   Lire (N4)
```

```
   Ecrire ("Entrer la valeur de la 5ème note")
   Lire (N5)
   Ecrire ("La note 1 multipliée par 3 est : ", N1 * 3)
   Ecrire ("La note 2 multipliée par 3 est : ", N2 * 3)
   Ecrire ("La note 3 multipliée par 3 est : ", N3* 3)
   Ecrire ("La note 4 multipliée par 3 est : ", N4 * 3)
   Ecrire ("La note 5 multipliée par 3 est : ", N5 * 3)
Fin
```

La même instruction se répète cinq fois. Imaginons que si l'on voudrait réaliser cet algorithme avec 100 notes, cela deviendrait fastidieux.

Comme les variables ont des noms différents, on ne peut pas utiliser de boucle, ce qui allonge considérablement le code et le rend très répétitif.

Pour résoudre ce problème, il existe un type de données qui permet de définir plusieurs variables de même type.

II. Définition

Un tableau est une suite d'éléments de même type. Il utilise plusieurs cases mémoire à l'aide d'un seul nom. Comme toutes les cases portent le même nom, elles se différencient par un numéro ou un indice.

Nous pouvons représenter schématiquement un tableau nommé **Note** composé de cinq cases, dans la mémoire comme suit :

	Note[1]	Note[2]	Note[3]	Note[4]	Note[5]	
Note	12	14	11.5	13	12	← Contenu du tableau

Nous disposons alors de cinq variables. Pour les nommer, on indique le nom du tableau suivi de son indice entre crochets ou entre parenthèses : La première s'appelle Note[1], la deuxième Note[2], etc. jusqu'à la dernière Note[5].

Note[4] représente le 4ème élément du tableau **Note** et vaut 13.

III. Tableau à une dimension

III.1. Déclaration

La déclaration d'un tableau permet d'associer à un nom une zone mémoire composée d'un certain nombres de cases mémoires de même type.

Syntaxe :

Variable identificateur : tableau[indice_min .. indice_max] de type
Ou bien
Variable identificateur : tableau[taille] de type

Notes :

- Le premier élément d'un tableau porte l'indice zéro ou l'indice 1 selon les langages.
- La valeur d'un indice doit être un nombre entier.
- La valeur d'un indice doit être inférieure ou égale au nombre d'éléments du tableau. Par exemple, avec le tableau tab[1 .. 20], il est impossible d'écrire tab[0] et tab[21]. Ces expressions font référence à des éléments qui n'existent pas.

Exemple :

L'instruction suivante déclare un tableau de 30 éléments de type réel :

```
Variable Note : tableau[1 .. 30] de Réels
```

Note : c'est le nom du tableau (identificateur).
1 : c'est l'indice du premier élément du tableau.
30 : c'est l'indice du dernier élément du tableau (nombre d'éléments du tableau).

Exercice :

Déclarer deux tableaux nommés A et B composé chacun d'eux de 10 éléments de type chaîne

Solution :

Prof. Omar El kharki & Mme Jamila Mechbouh / elkharki@gmail.com

Variables A, B : tableau[1 .. 10] de chaîne

Remarques :

- Lorsqu'on déclare un tableau, on déclare aussi de façon implicite toutes les variables indicées qui le constituent. Nous utiliserons souvent la valeur 1 comme premier indice mais on peut aussi utiliser un autre indice minimum, comme 0. Dans ce cas, l'indice maximum sera égal au nombre d'éléments – 1.
- Dans le langage C, les cases du tableau sont numérotées à partir de 0. En Visual Basic l'indice minimum est 1.

III.2. Utilisation

Les éléments d'un tableau sont des variables indicées qui s'utilisent exactement comme n'importe quelles autres variables classiques. Elles peuvent faire l'objet d'une affectation, elles peuvent figurer dans une expression arithmétique, dans une comparaison, elles peuvent être affichées et saisies etc.

L'utilisation de ces éléments se fait en suite, via le nom du tableau et son indice. Ce dernier peut être soit une valeur (exemple : Note[3]) , soit une variable (exemple : Note[i]) ou encore une expression (exemple : Note[i+1]).

Remarque :

Il ne faut pas confondre l'indice d'un élément d'un tableau avec son contenu. Par exemple, la 4eme maison de la rue n'a pas forcément 4 habitants.

Exemple 1 :

L'instruction suivante affecte à la variable X la valeur du premier élément du tableau Note :

X ← Note[1]

Exemple 2 :

L'instruction suivante affiche le contenu du quatrième élément du tableau Note :

Prof. Omar El kharki & Mme Jamila Mechbouh / elkharki@gmail.com

```
Ecrire(Note[4])
```

Exemple 3 :

L'instruction suivante affecte une valeur introduite par l'utilisateur à l'élément trois du tableau Note

```
Lire (Note[3])
```

➢ Parcours complet d'un tableau :

Le fait que les éléments d'un tableau soient indicés permet de parcourir tous ces éléments avec une boucle, on utilisant une variable qui sert d'indice et s'incrémente à chaque tour de boucle.

Exercice 1 :

Ecrire un algorithme permettant de saisir 30 notes et de les afficher après avoir multiplié toutes ces notes par un coefficient fourni par l'utilisateur :

Solution :

```
Algorithme  tableau_note
Variable Note : tableau[1..30] de Réels
            Coef, i : entier
Début
   Ecrire ("Entrer le coefficient")
   Lire (Coef)
   // Remplissage du tableau Note
   Pour i ← 1 jusqu'à 30 Faire
      Ecrire ("Entrer la valeur de la note", i)
      Lire (Note[i])
   FinPour
   // Affichage des notes * Coef
   Pour i ← 1 jusqu'à 30 Faire
      Ecrire (Note[i] * Coef)
   FinPour
Fin
```

Exercice 2 :

Prof. Omar El kharki & Mme Jamila Mechbouh / elkharki@gmail.com

Ecrire un algorithme qui calcul la somme des éléments d'un tableau.

Solution :

On suppose que le nombre d'éléments du tableau ne dépasse pas 100.

```
Algorithme  Somme_tableau
Constante M = 100
Variable T : tableau[1..M] de réels
            n, i : entier
            s : réel
Début
   // Demande de la taille du tableau
   Ecrire ("Entrer la taille du tableau")
   Lire (n)
   Si n > M alors
      N ← M
   FinSI
   // Remplissage du tableau
   Pour i ← 1 jusqu'à n Faire
      Ecrire ("Entrer la valeur de l'élément", i, "du tableau")
      Lire (T[i])
   FinPour
   // Calcul la somme  des éléments du tableau
   s ← 0
   Pour i ← 1 jusqu'à n Faire
      s ← s + T[i]
   FinPour
   Ecrire ("La somme des éléments du tableau est :", s)
Fin
```

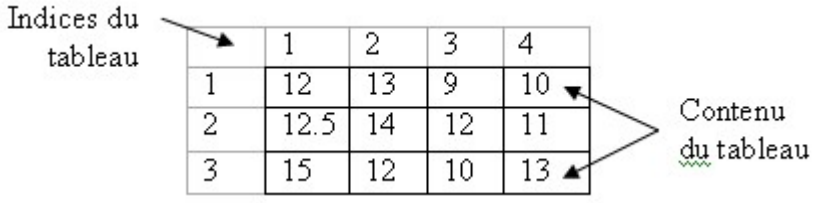
IV. Tableau à deux dimensions

Reprenons l'exemple des notes en considérant cette fois qu'un étudiant a plusieurs notes (une note pour chaque matière). Pour quatre étudiants, nous aurions le tableau de relevés des notes suivant :

	Etudiant 1	Etudiant 2	Etudiant 3	Etudiant 4
Informatique	12	13	9	10
Comptabilité	12.5	14	12	11
Mathématiques	15	12	10	13

Les tableaux à deux dimensions se représentent comme une matrice ayant un certain nombre de lignes (première dimension) et un certain nombre de colonnes (seconde dimension).

Nous pouvons représenter schématiquement un tableau de 3 lignes et de 4 colonnes comme suit :



IV.1. Déclaration

Syntaxe :

Variable identificateur : tableau[1..nb_lignes, 1..nb_colonnes] de type
Ou bien
Variable identificateur : tableau[nb_lignes, nb_colonnes] de type

Exemple :

L'instruction suivante déclare un tableau Note de type réel à deux dimensions composé de 3 lignes et de 4 colonnes :

```
Variable Note : tableau[1..3, 1..4] de réels
```

IV.2. Utilisation

Pour accéder à un élément de la matrice (tableau à deux dimensions), il suffit de préciser, entre crochets, les indices de la case contenant cet

Prof. Omar El kharki & Mme Jamila Mechbouh / elkharki@gmail.com

élément.

Les éléments de la matrice peuvent être utilisés comme n'importe quelle variable.

Exemple :

L'instruction suivante affecte à la variable X la valeur du premier élément du tableau Note :

X ← Note[1,1]

➢ **Parcours complet d'un tableau à deux dimensions:**

Pour parcourir une matrice nous avons besoin de deux boucles, l'une au sein de l'autre, c'est ce qu'on appelle les boucles imbriquées. La première boucle par exemple est conçue pour parcourir les lignes tandis que la deuxième est utilisée pour parcourir les éléments de la ligne précisée par la boucle principale (la première boucle).

Exercice :

Ecrire un algorithme permettant la saisie des notes d'une classe de 30 étudiants en 5 matières.

Solution :

Prof. Omar El kharki & Mme Jamila Mechbouh / elkharki@gmail.com

```
Algorithme  Notes
Constante N = 30
          M = 5
Variable note : tableau[1..N, 1..M] de Réels
        i, j : entier
Début
  // Remplissage du tableau note
  Pour i ← 1 jusqu'à N Faire
    Pour j ← 1 jusqu'à M Faire
      Ecrire ("Entrer la note de l'étudiant", i, "dans la matière", j)
      Lire (note[i,j])
    FinPour
  FinPour
Fin
```

V. Tableau dynamique

Les tableaux définis jusqu'ici sont dits statiques, car il faut qu'au moment de l'écriture du programme le programmeur décide de la taille maximale que pourra atteindre le tableau. Si le programmeur donne une taille très grande alors que lui il n'a pas besoin que d'une petite taille, dans ce cas le programme consomme trop de mémoire. Dans certaines situations, on ne peut pas savoir la taille du tableau dans la phase de programmation.

Les tableaux dynamiques sont des tableaux dont la taille n'est définie que lors de l'exécution. Pour créer un tableau dynamique, il suffit de lui affecter une taille vide.

Syntaxe :

Variable identificateur : **tableau** [] de type

Comme un tableau dynamique ne possède pas de taille prédéfinie, il convient de redimensionner le tableau avant de pouvoir s'en servir.

Syntaxe :

Redimensionner identificateur[N]

Prof. Omar El kharki & Mme Jamila Mechbouh / elkharki@gmail.com

Exemple :

```
Variable t : tableau[] d'entiers
Redimensionner t[11]
```

La première instruction déclare un tableau dynamique. La deuxième redimensionne la taille du tableau t à 11 éléments.

Note : Tableau dynamique en VB

Pour créer un tableau dynamique d'entiers en VB on utilise la syntaxe suivante :

Dim T() As Integer

L'instruction suivante redimensionne le tableau T en 11 éléments:

Redim T(11)

Vous pouvez utiliser l'instruction ***ReDim*** pour changer la taille d'un tableau qui a déjà été déclaré. Si vous disposez d'un grand tableau et que certains de ses éléments ne sont plus nécessaires, ***ReDim*** peut libérer de la mémoire en réduisant la taille du tableau. En revanche, si votre code détermine qu'un tableau doit contenir plus d'éléments, ***ReDim*** peut les ajouter.

A chaque appel de ***ReDim***, le contenu du tableau précédent est effacé. Pour conserver les valeurs existantes, il faudra rajouter le mot clé ***Preserve***.

Exemple :

```
Redim Preserve T(11)
```


Prof. Omar El kharki & Mme Jamila Mechbouh / elkharki@gmail.com