Chapitre

Les tableaux et les chaînes de caractères

Objectif

Souvent, il est nécessaire de mémoriser une suite de valeurs numériques, de mots ou de phrases dans des variables. Mais, utiliser autant de variables n'est réellement rationnel, donc ce qui est alors pratique est de créer des tableaux ou des chaînes de caractères, ce qui allégera les déclarations des variables et simplifiera leur lecture, leur écriture et leur accès.

Introduction

Imaginons que l'on veut calculer la moyenne des notes d'une promotion mais en gardant en mémoire toutes les notes des étudiants (pour par exemple faire d'autres calculs tels que l'écart type, la note minimale, la note maximale, le classement des étudiants, etc.).

Il faudrait définir autant de variables individuelles et assurer leurs gestions.

Cette solution est possible en théorie mais impossible en pratique car l'implémentation est très complexe notamment au niveau des tâches nécessitant la manipulation de l'ensemble de ces variables (parcours, recherche, tri, ...).

Cette solution est possible en théorie mais impossible en pratique car l'implémentation est très complexe notamment au niveau des tâches nécessitant la manipulation de l'ensemble de ces variables (parcours, recherche, tri, ...).

De ce fait, la solution possible pour ce problème est de :

- Définir une variable agrégeant un grand nombre de sous-variables de même type élémentaire
- Autoriser l'utilisation d'une sous-variable individuelle (du type élémentaire) au sein de cette variable par la donnée d'un indice (ou de plusieurs indices)

C'est le rôle des tableaux

1. Le type tableau

Un "Tableau" (array en anglais) est une variable agrégeant un nombre arbitraire N de sous-variables ("composantes", "éléments") de même type [13]. C'est une structure de donnée T qui permet de stocker un certain nombre d'éléments T[i] repérés par un index i.

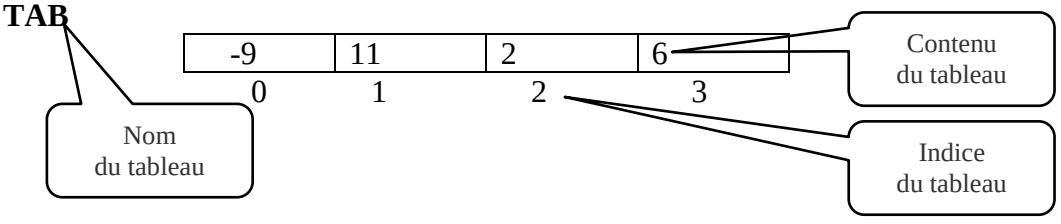

Figure 8. Structure d'un tableau

Les tableaux vérifient généralement les propriétés suivantes :

- Tous les éléments ont le même type de base ;
- Le nombre d'éléments stockés est fixé

- On appelle souvent vecteur un tableau en une dimension.
- La taille de la séquence est constante ; elle est spécifiée une fois pour toute lors de la déclaration du tableau, Le premier index d'un tableau de N éléments est 0 et le dernier index est N -1.

1.1. Manipulation d'un tableau

Pour bien utiliser un tableau on doit maitriser les tâches de déclaration, d'accès, de chargement et d'affichage.

1.1.1. Déclaration

En algorithmique la déclaration d'un tableau est donnée par la ligne suivante :
Non du tableau : Tableau [nombre d'éléments] type éléments

Exemple :

T1 : Tableau [50] d'entier
T2 : Tableau [20] de réel
T3 : Tableau [30] de caractère

En langage C

Type Nom du Tableau [nombre d'éléments];

Exemple :

```
int tableau [4];
```

On déclare ici un tableau contenant 4 valeurs de type int. "[4]" représente le nombre de cases du tableau, on aura donc ici un tableau de type int de 4 cases, ces cases étant remplies de données de type **int uniquement**.

De plus la taille du tableau doit être connue à la compilation, et on évitera par-dessus tout de définir sa taille à partir d'une variable. Ainsi :

```
int variable = 4;
int tableau [variable]; n'est pas valable
```

1.1.2. Accès aux composantes d'un tableau

Considérons un tableau T de dimension N

- L'accès au premier élément du tableau se fait par **T [0]**
- L'accès au dernier élément du tableau se fait par **T [N-1]**
- L'accès à l'élément avec l'indice i se fait par T[i]

1.1.3. Chargement d'un tableau

Le chargement d'un tableau se fait généralement ave une boucle POUR :

Algorithme	***Langage C***

Variables
T : Tableau [N] d'entiers
i : entier
DÉBUT
Pour i =1**A** N**Faire**
Écrire ('T ['', i, ''] :')
Lire (T[i])
Fin pour
FIN
	#include<stdio.h>
	int main ()
	{
	int TAB[5],i;
	for (i=0;i<=4;i++)
	{
	printf("TAB[%d] = ",i);
	scanf("%d",&TAB[i]);
	}
	return 0;
	}

1.1.4. Affichage d'un tableau

Algorithme	***Langage C***

Variables
T :Tableau[N] d'entiers
i : entier
DÉBUT
Pour i =1**A** N1
Faire
Ecrire (T[i])
Fin pour
FIN
	#include<stdio.h>
	int main ()
	{
	int TAB[5],i;
	for (i=0;i<=4;i++)
	{
	printf("TAB[%d] = ",i);
	}

1.1.5. Initialisation d'un tableau

Il est possible d'initialiser un tableau avec une liste d'expressions constantes séparées par des virgules, et entourée des signes {et}. Exemple :

int TAB [5] = {1, 2, 3, 4, 5};

On peut donner moins d'expressions constantes que le tableau ne comporte d'éléments. Dans ce cas, les premiers éléments du tableau seront initialisés avec les valeurs indiquées, les autres seront initialisés à zéro. Exemple :

int TAB[5] = {1, 2};

Les éléments d'indice 0 et 1 seront initialisés respectivement avec les valeurs 1 et 2, les autres éléments seront initialisés à zéro.

2. Les tableaux multidimensionnels

Dans cette section on ne garde que les tableaux à deux dimensions (matrices) vu leur utilité pratique, notamment en mathématiques.

Une matrice est un ensemble de données de même type logées en mémoire centrale et référencé par deux indices (les lignes et les colonnes).Une matrice est caractérisée par:

- Le nom
- Le nombre de colonne
- Le nombre de ligne
- La taille de la matrice (nombre de ligne x nombre de colonnes)
- Le type des éléments de la matrice.

Chaque élément dans une matrice est caractérisé par le numéro de la ligne et le numéro de la colonne.

$$M = \begin{pmatrix} 1 & 2 & 0 \\ 9 & 7 & 4 \end{pmatrix}$$

M[1,1] = 1 , M[1,3] = 0 , M[2,2] = 7 …etc.

2.1. Déclaration

Nom-MTRICE = tableau [Nbr-Lignes, Nbr-Colonnes] de type élément

En langage C : int tab [M] [N] ;

Exemple:
Matrice = tableau[3, 4] de entier
int Matrice [3] [4] ;

2.2. La lecture d'une matrice

Variable
MAT: tableau [n, m] d'entier
i, j: entier
Début
Pour i=1**À** n **Faire**
Pour j=1À m **Faire**
lire (MAT[i, j])
Fin Pour
Fin Pour
Fin
	#include<stdio.h>
	intmain()
	{
	int MAT[3][2] ;
	int i, j;
	for (i = 0 ; i < 3; i++)
	{
	for (j = 0; j < 2; j++)
	{
	printf("MAT [%d,%d]= ",i,j);
	scanf("%d",&MAT[i][j]);
	}
	}
	return 0;
	}

Pour lire colonne par colonne, on inverse juste i et j

2.3. Ecriture d'une matrice

Variable
MAT: tableau[n,m] d'entier
i, j: entier
Début
Pour i=1**À** n1**Faire**
Pour j=1**À** m**Faire**
Ecrire (MAT [i, j])
Fin Pour
Fin Pour
Fin
	#include<stdio.h>
	intmain()
	{
	inttab[3][2] ;
	int i, j;
	for (i = 0 ; i < 3; i++)
	{
	for (j = 0; j < 2; j++)
	{
	printf("tab[%d,%d]= ",i,j);
	scanf("%d",&tab[i][j]);
	}
	}
	for (i = 0 ; i < 3; i++)
	for (j = 0; j < 2; j++)
	printf("tab[%d,%d]= %d \n",i,j,tab[i][j]);
	return 0;
	}

2.4. Initialisation d'une matrice

Lors de la déclaration d'un tableau, on peut initialiser les composantes du tableau, en indiquant la liste des valeurs respectives entre accolades. A l'intérieur de la liste, les composantes de chaque ligne du tableau sont encore une fois comprises entre accolades. Pour améliorer la lisibilité des programmes, on peut indiquer les composantes dans plusieurs lignes.

Exemples :

```
int A[3][10] ={{ 0,10,20,30,40,50,60,70,80,90},
               {10,11,12,13,14,15,16,17,18,19},
               { 1,12,23,34,45,56,67,78,89,90}};

floatB[3][2] = {{-1.05,   -1.10  },
                {86e-5,    87e-5 },
                {-12.5E4, -12.3E4}};
```

Lors de l'initialisation, les valeurs sont affectées ligne par ligne en passant de gauche à droite. Nous ne devons pas nécessairement indiquer toutes les valeurs: Les valeurs manquantes seront initialisées par zéro. Il est cependant défendu d'indiquer trop de valeurs pour un tableau.

2.5. Accès à un tableau à deux dimensions

En algorithmique :

<NomTableau>[<Ligne>, <Colonne>]

En langage C

```
<NomTableau>[<Ligne>][<Colonne>]
```

Les éléments d'un tableau de dimensions L et C se présentent de la façon suivante:

```
|A[0][0]    A[0][1]    A[0][2]    . . .    A[0][C-1]  |
|A[1][0]    A[1][1]    A[1][2]    . . .    A[1][C-1]  |
|A[2][0]    A[2][1]    A[2][2]    . . .    A[2][C-1]  |
|   . . .      . . .      . . .     . . .      . . .  |
|A[L-1][0]  A[L-1][1]  A[L-1][2]   . . .   A[L-1][C-1]|
```

Considérons un tableau A de dimensions L et C. En C,

- les indices du tableau varient de 0 à L-1, respectivement de 0 à C-1.
- la composante de la Nième ligne et Mième colonne est notée: **A[N-1][M-1]**

En langage algorithmique,

- les indices du tableau varient de 1 à L, respectivement de 1 à C.
- la composante de la Nième ligne et Mième colonne est notée: **A[N,M]**

3. Les chaînes de caractères

Les chaînes de caractères est un cas particulier des tableaux de caractères ou se dernier peut être initialisé par une liste de constantes caractères. Exemple : charch[3] = {'a',

'b', 'c'};. C'est évidemment une méthode très lourde. Dans ce cas Un tableau de caractères peut être initialisé par une chaîne littérale.

Exemple :

charch[8] = "exemple";

On se rappelle que le compilateur complète toute chaîne littérale avec un caractère nul, il faut donc que le tableau ait au moins un élément de plus que le nombre de caractères de la chaîne littérale.

Il est admissible que la taille déclarée pour le tableau soit supérieure à la taille de la chaîne littérale.

Exemple :

char ch[100] = "exemple";

Dans ce cas, seuls les 8 premiers caractères de **ch** seront initialisés.

Il est également possible de ne pas indiquer la taille du tableau et dans ce cas, le compilateur à le bon goût de compter le nombre de caractères de la chaîne littérale et de donner la taille adéquate au tableau (sans oublier le *null*).

Exemple :

char ch[] = "ch aura 22 caractères";

Il est également possible de donner au tableau une taille égale au nombre de caractères de la chaîne. Dans ce cas, le compilateur comprend qu'il ne faut pas rajouter le *null* de la fin de chaîne. Exemple :

char ville[7] = "Exemple";

L'accès à un élément d'une chaîne de caractères peut se faire de la même façon que l'accès à un élément d'un tableau. En déclarant une chaîne par:

char A[6];

Nous avons défini un tableau A avec six éléments, auxquels on peut accéder par:

A[0], A[1], ... , A[5]

Conclusion

Nous avons vu à travers ce chapitre comment utiliser des tableaux et des chaînes de caractères. Dans la pratique, les types de tableaux les plus utilisés sont les vecteurs (tableaux à une seule dimension) et les matrices (tableaux à deux dimensions).