CHAPITRE 2 : LES TABLEAUX

I. Définitions :

Les variables que nous avons utilisés jusqu'à présent étaient ce que l'on nomme des variables simples : à un instant donné, une variable de ce type contient une seule valeur.

*Un **tableau** représente sous un même nom une liste de variables de même type, numérotées par un ou plusieurs indices. Les indices permettent de désigner séparément chaque variable (en donnant le nom du tableau et les numéros d'indices correspondant à cette variable).*

*Un **vecteur** ou **tableau à une dimension** ne possède qu'un seul indice.*
*Une **matrice** ou **tableau à deux dimensions** possède deux indices.*

Du point de vue de sa représentation, un tableau utilise une zone contiguë de la mémoire : L'adresse d'une variable particulière du tableau peut alors facilement être calculée à partir de l'adresse de départ du tableau, des indices de la variable et de la taille de chaque variable.

Un tableau a donc *une taille fixe*, car il est difficile de faire varier dynamiquement la taille d'une zone mémoire contiguë.

Les tableaux permettent donc de gérer un ensemble de variables dont on connaît le nombre a priori et auxquels on veut pouvoir accéder directement par l'intermédiaire d'un indice. Par contre, l'insertion de nouveaux éléments ou la destruction d'anciens est difficile à réaliser.

II. Les tableaux à une dimension :

II-1. Déclaration :

Comme toute variable, un tableau soit faire l'objet d'une déclaration :

```
int tableau[10];
```

L'instruction ci-dessus crée un tableau de 10 entiers. Il est possible d'initialiser un tableau au moment de sa déclaration par :

```
int tableau[10]={1,2,3,4,5,6,7,8,9,10};
```

Dans le cas de l'initialisation à la déclaration ; si on ne fournit pas suffisamment d'éléments pour remplir toutes les cases du tableau, les cases restantes sont remplies avec la valeur 0. Ainsi, les deux instructions suivantes sont équivalentes :

- int tab1[10]={1,2,3,4,0,0,0,0,0,0};
- int tab1[10]={1,2,3,4};

De la même manière un tableau de caractères peut être initialisé par une liste de caractères, mais aussi par une chaîne de caractères littérale. Notons que le compilateur complète toute chaîne de caractères avec un caractère nul '\0'. Il faut donc que le tableau ait au moins un élément de plus que le nombre de caractères de la chaîne littérale.

Exemple :

```
main()
{
char tab[8] = "exemple";
int i;
for (i = 0; i < 7; i++)
printf("tab[%d] = %c\n",i,tab[i]);
}
```

Lorsqu'un tableau est initialisé au moment de sa déclaration, il est inutile de préciser son nombre d'éléments, qui sera déterminé automatiquement à la compilation. Il suffit simplement de ne pas mettre de dimension entre les **[ ]** habituels. Par exemple, les deux déclarations suivantes sont équivalentes :

- int tab1[4]={1,2,3,4};
- int tab1[]={1,2,3,4};

II-2. Accès aux éléments d'un tableau :

Pour accéder aux éléments d'un tableau, on peut préciser l'indice de l'élément qui nous intéresse. Le premier élément d'un tableau porte l'indice 0. Par exemple, en reprenant la variable tableau précédente, tab[0] vaut 1, et tab[3] vaut 4.

Exemples :

```
/* Afficher un tableau de taille n=5:*/
#include<stdio.h>
main()
{
int A[5]={1,6,12,19,-2};
 int i;
for (i=0; i<5; i++)
                printf(" Le contenu de la case %d est %d ",i, A[i]);

}
```

```
/* Remplir un tableau de taille n=5:*/

#include<stdio.h>
main()
{
int A[5];
 int i;
for (i=0; i<5; i++) {
                printf("Entrer la valeur de la case %d ",i);
                scanf("%d ",&A[i]);
                        }
}
```

III. Les tableaux à deux dimensions (Matrices):

Il est possible de créer des tableaux de tableaux, ce qui revient à définir des tableaux à plusieurs dimensions. Voici un exemple de déclaration d'un tableau de réels à 4 lignes et 5 colonnes :

float tableau[4][5];

Il n'y a pas de limite (à part la mémoire disponible) au nombre de dimensions d'un tableau.
Si le tableau est initialisé lors de sa déclaration ...

int tableau[2][3]={{1,2,3},{4,5,6}};

il est alors rempli de la façon suivante :

1	2	3
4	5	6

Exemples :

```
int compteur[4][5];
float nombre[2][10];
int A[3][2] = {{1, 2 }, {10, 20 }, {100, 200}};
```

/* Afficher un tableau à deux dimensions:*/

```
#include<stdio.h>
main()
{
int A[5][10];
int I,J;
        /* Pour chaque ligne ... */
        for (I=0; I<5; I++)
        {
                for (J=0; J<10; J++)
                        printf("%d\t", A[I][J]);

                printf("\n"); /* Retour à la ligne suivante */
        }

 }
```

/* Remplir un tableau à deux dimensions:*/

```
#include<stdio.h>
main()
{
int A[5][10];
int I,J;

        for (I=0; I<5; I++)
         {
                for (J=0; J<10; J++)
                {
                        printf("Entrer la valeur de la case %d,%d", I,J);
                        scanf("%d", &A[I][J]);
                }
        }
}
```