Les tableaux

DAKKAR Borhen-eddine

Lycée le Corbusier

BTS SN-IR

Table des matières

Table des matières

Les tableaux unidimensionnels

- Les variables que nous avons utilisé jusqu'à présent ont toutes une caractéristique commune : chaque variable ne pouvait être utilisé que pour stocker une seule valeur à la fois.
- Souvent, nous pouvons avoir un ensemble de valeurs, toutes du même type de données, qui forment un groupe. Par exemple, la figure ci-dessous illustre trois groupes d'éléments.

Températures	**Voltages**	**codes**
35.4	5	A
37.7	12	B
38.2	24	C
38.1	48	D

- Nous allons voir comment les tableaux unidimensionnels sont déclarés, initialisés, stockés dans un ordinateur et utilisés.

Déclaration des tableaux unidimensionnels

- Le langage C permet d'utiliser des tableaux. On nomme ainsi un ensemble d'éléments de même type désignés par un **identificateur** unique; chaque élément est repéré par un **indice** précisant sa position au sein du tableau.
- Regardons les températures de la liste prcédente, nous pouvons voir qu'elles sont des valeurs à virgule flottante.
- Nous choisissons **Temp** comme identifcateur de notre tableau. Pour préciser que **temp** va stocker cinq valeurs à virgule flottante individuelles, nous devons faire la déclaration float temp [3].
- De même pour les autres données :

```
int volts[3];
char code[3];
```

- Chaque tableau dispose de suffisamment de mémoire réservée pour contenir le nombre des éléments donnés dans la déclaration.

Déclaration des tableaux unidimensionnels

- Les éléments du tableau sont stockés séquentiellement, avec le premier élément de tableau dans le premier emplacement réservé, le second élément dans le deuxième emplacement réservé, et ainsi de suite jusqu'à ce que le dernier élément soit stocké dans le dernier emplacement réservé.
- Pour accéder à un élément dans un tableau, il faut donner l'identificateur du tableau et la position de l'élément. Cette position est appelée **indice**. Conventionnellement, en langage C, la première position porte le numéro 0.
- Par exemple :

```
temp[0]; //fait référence à la 1ière température dans le tableau.
temp[1]; //fait référence à la 2ème température dans le tableau.
temp[2]; //fait référence à la 3ième température dans le tableau.
temp[3]; //fait référence à la 4ième température dans le tableau.
```

Quelques règles pour les tableaux

- L'écreture : temp[0] = 35.4; veut dire que nous affectons la valeur 35.4 à l'élément 0 de nôtre tableaux **temp**.
- L'élément d'un tableau peut aussi apparaître comme opérande d'un opérateur d'incrémentation, comme dans :

```
t[3]++;
--t[i];
```

- L'indice contenu entre crochets n'a pas besoin d'être une constante entière; toute expression qui résulte en un entier peut être utilisée comme indice.

```
temp[i];
temp[2*i];
temp[j-i];
```

Entrées/sorties d'un tableau

- Des valeurs individuelles peuvent être attribuées à des éléments de tableau en utilisant fonction **scanf()**.

```
scanf("%f", &temp[0]);
scanf("%f %f %f", &temp[1], &temp[2], &temp[3]);
```

Dans la première instruction, une seule valeur sera lue et stockée dans la variable nommé **temp[0]**. La deuxième instruction entraîne la lecture et le stockage de trois valeurs dans les variables **temp[1]**, **temp[2]** et **temp[3]**, respectivement.

- Alternativement, une boucle for peut être utilisée pour parcourir le tableau pour une entrée de données interactive. Par exemple :

```
for(int i=0, i<=3, i++)
    {
    printf("Entrer la température :");
    scanf("%f", &temp[i]);
    }
```

Entrées/sorties d'un tableau

- Les éléments d'un tableau peuvent être affichés à l'aide de **printf ()** séprément. Il est possible aussi d'afficher tous les éléments en utilisant une boucle **for**.

```
printf("%f", temp[2]);
for(int i=0, i<=3, i++)
    {printf("La température de lélément %d est %f:", i,temp[i]);}
```

- Il est possible d'initialiser un tableau au moment de sa décalaration :

```
    float temp[3] = {35.4, 37.7, 38.2, 38.1};
```

- Il est possible d'omettre la dimension du tableau, celle-ci étant alors déterminée par le compilateur par le nombre de valeurs énumérées dans l'initialisation. Par exemple :

```
    char codes [] = {'s', 'a', 'm', 'p','l', 'e'};
```

réserve six emplacements de caractères pour un tableau nommé **codes**.

Exemple

Le tableau **notes** stocke cinq nombres entiers. Nous utilisons deux boucles **for**, la première pour parcourir chaque élément du tableau et saisir sa valeur, la deuxième pour afficher ces valeurs.

```c
#include <stdio.h>
#include <stdlib.h>
main( )
{
int i, Notes[5];
for (i = 0; i <= 4; ++i)/* Entrer 5 notes */
{
    printf("Entrer une note : ");
    scanf("%d", &Notes[i]);
}
for (i = 0; i <= 4; ++i)/* Afficher 5 notes */
{
    printf("\nNotes[%d] is %d", i, grades[i]);
}
}
```

Table des matières

Tableaux bidimensionnels

- Un tableau à deux dimensions, également appelé tableau, se compose de lignes et de colonnes d'éléments. Par exemple :

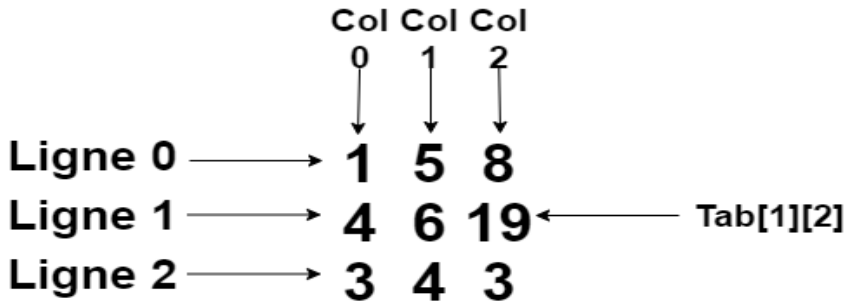
$$\begin{matrix} 1 & 5 & 8 \\ 4 & 6 & 19 \\ 3 & 4 & 3 \end{matrix}$$

  est un tableau compsé de trois lignes et trois colennes.
- Pour réserver de l'espace mémoire pour ce tableau, le nombre de lignes et de colonnes doit être inclus dans la déclaration du tableau.

```
int Tab [2][2];
```

- Chaque élément dans un tableau à deux dimensions est identifié par sa position dans le tableau.
- Tab [1] [2] identifie l'élément de la ligne 1, colonne 2.

Tableaux bidimensionnels



- Les tableaux bidimensionnels peuvent être initialisés dans leurs déclarations. Cela se fait en listant les valeurs initiales entre accolades et en les séparant par des virgules.

```
int Tab [3][3] = {{1, 5, 8},{4, 6, 19},{3, 4, 3}};
```

- La séparation des valeurs en lignes dans l'instruction de déclaration n'est pas nécessaire puisque le compilateur attribue des valeurs commençant par l'élément [0][0] et continue ligne par ligne pour remplir les valeurs restantes. Ainsi, la déclaration peut être :

```
int Tab [3][3] = {1, 5, 8, 4, 6, 19, 3, 4, 3};
```

Exemple

```c
#include <stdio.h>
#include <stdlib.h>
main( )
{
int i, j, int Tab [3][3] = {1, 5, 8, 4, 6, 19, 3, 4, 3};
printf("\n Affichage des éléments du tableau Tab");
printf ("\n %2d %2d %2d %2d",Tab [0][0],Tab [0][1],Tab [0][2]);
printf ("\n %2d %2d %2d %2d",Tab [1][0],Tab [1][1],Tab [1][2]);
printf("\n %2d %2d %2d %2d",Tab [2][0],Tab [2][1],Tab [2][2]);
printf("\n Affichage avec des boucles for imbriquées");
for (i = 0; i < 3; ++i)
{
    printf (" \n") ; /* saut de ligne pour chauqe nouvelle ligne */
    for (j = 0; j < 3; ++j)
    {printf ("%2d ", Tab [i][j]);}
}
}
```

Références