

Les tableaux

slides julien.brajard@upmc.fr revus par maria.potop-butucaru@lip6.fr

Polytech'UPMC

Octobre 2018

Généralités

- Un tableau est un ensemble d'éléments de même type désigné par un identificateur unique.
- Chaque élément est repéré par un **indice** précisant sa position au sein de l'ensemble.
- Il se déclare en même temps que les autres variables.
- Il existe deux grandes familles de tableaux :
 - ▶ Les tableaux unidimensionnels (vecteurs)
 - ▶ Les tableaux multidimensionnels (matrices)

Déclaration de tableaux unidimensionnels

- On doit réserver un emplacement mémoire pour un certain nombre d'éléments.

```
type nom_tableau [ nb_elements ] ;
```

- ▶ *type* est le type des éléments du tableau *nom_tableau*
 - ▶ *nb_elements* est le nombre d'éléments que contient le tableau.
- Conventionnellement, **la première position porte le numéro 0**. Les indices vont donc de 0 à *nb_elements*-1

```
int tab[10] ; // tab est un tableau de 10 entiers  
float zz[5] ; // zz est un tableau de 5 réels  
char y[12] ; // y est un tableau de 12 caractères
```

Quelques règles

- Chaque élément est stocké dans une case d'un tableau (localisé par un **indice**).
- Un élément de tableau est une variable :
 - ▶ On peut l'initialiser.
 - ▶ On peut l'utiliser dans une expression.
- Le premier élément du tableau est à l'indice **0**.
- La dimension d'un tableau (nombre d'éléments) ne peut être qu'une constante ou une expression constante (pas une variable).
- Il faut faire très attention aux problèmes de débordement d'indice (cas où l'on veut accéder à un indice de tableau supérieur au nombre d'éléments prévus).

Initialisation des éléments (1)

- Initialisation de l'élément d'indice k :

```
nom_tableau[ $k$ ] = valeur ;  
nom_tableau[ $k$ ] = expression ;
```

```
tab[0]=5 ;  
/* le premier élément du  
tableau tab a pour  
valeur 5*/
```

- Initialisation à la déclaration

```
//Place les valeurs 1,2,3,4,5 dans chacun des  
//5 éléments du tableau :  
int tab[5] = {1,2,3,4,5};
```

```
//Ne remplit que certains indices (ici 2 et 4) :  
int tab[5] = {, ,3, ,5};
```

```
//Ne remplit que les 2 premiers indices (0 et 1) :  
int tab[5] = {2,3};
```

```
//Determine automatiquement la taille du tableau (en fonction  
//du nombre de valeurs :  
int tab[] = {1,2,3,4,5};
```

Initialisation des éléments (2)

Initialisation des éléments dans une boucle `for`

0	0	0	0	0
---	---	---	---	---

0	2	4	6	8
---	---	---	---	---

```
int j ; //ind. de boucle
int Tab[5] ;

for (j = ... ; ... ; ...)
{
    ...
}
```

```
int j ; //ind. de boucle
int Tab[5] ;

for (j = ... ; ... ; ...)
{
    ...
}
```

Un exemple (version 1)

```
#include <stdio.h>

/* Remplissage d'un tableau de
notes et calcul de la moyenne */

int main()
{
    float notes[50], moy=0.0;
    int i;

    for (i=0;i<50;i++)
    {
        printf("Entrez la note: ");
        scanf("%f",&notes[i]);
        moy += notes[i];
    }
    printf("moyenne = %f",moy/50);
}
```

- On remplit les 50 éléments du tableau (utilisation d'une boucle `for`)
- `i` sert d'indice pour parcourir les éléments du tableau.
- La moyenne est calculée en ajoutant à chaque passage de boucle, la nouvelle note saisie au clavier (`scanf`)

Problèmes dans l'exemple

- Utilisation non sécurisée de `scanf`
 - ▶ Vérification du nombre d'éléments correctement entrés
 - ▶ "Nettoyage de la mémoire tampon".

Voir le cours n°4 sur les entrées-sorties

- Problème pratique si on veut modifier la taille du tableau.

Un exemple (version 2)

```
#include <stdio.h>
#define NMAX 50

/* Remplissage d'un tableau de
notes et calcul de la moyenne */

int main()
{
    float notes [NMAX], moy=0.0;
    int i;

    for (i=0;i<NMAX;i++)
    {
        printf("Entrez la note: ");
        scanf("%f",&notes[i]);
        moy += notes[i];
    }
    printf("moyenne = %f",moy/NMAX);
}
```

Définition de la taille du tableau par un `#define` (fortement conseillé).

Si la valeur de `NMAX` est modifiée, il faut recompiler.

Les tableaux multidimensionnels

Déclaration

```
type nom_tableau [dim1][dim2]...[dimN];
```

Initialisation

```
//tableau vu comme 5 tableaux de deux éléments chacun
```

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

```
//Eléments rangés en mémoire automatiquement :
```

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

```
//Omission de valeurs :
```

```
int tab[5][2] = {1,,3,,,7,8,9,};
```

```
//Accès à un indice (après déclaration):
```

```
tab[3][0] = 12 ;
```

Algorithmes classiques sur les tableaux

- Initialisation d'un tableau à une valeur constante.
- Copie d'un tableau.
- Vérification de l'égalité de deux tableaux.
- Recherche d'un élément dans un tableau.
- Comptage du nombre d'occurrences d'un élément dans un tableau.
- Tri des éléments d'un tableau.