Les Tableaux

Pr Mostafa SAADI – ENSAK

L'Ecole Nationale des Sciences Appliquées à Khouribga

saadi_mo@yahoo.fr

9 octobre 2019

Plan du Cours

Introduction

Définitions

1. Il s'agit de variables (caractères, entiers, réels, etc.) stockées **consécutivement** dans la mémoire et que l'on peut manipuler **globalement** ou bien **élément par élément**.
2. Un **tableau** est un **espace réservé en mémoire** pour stocker une série d'**éléments de même type**.

Définition d'un tableau

Un **tableau** est une structure de donnée qui permet de stocker un **nombre fini d'éléments**, de **même type de base**, repérés par un index. Les tableaux vérifient généralement les propriétés suivantes :

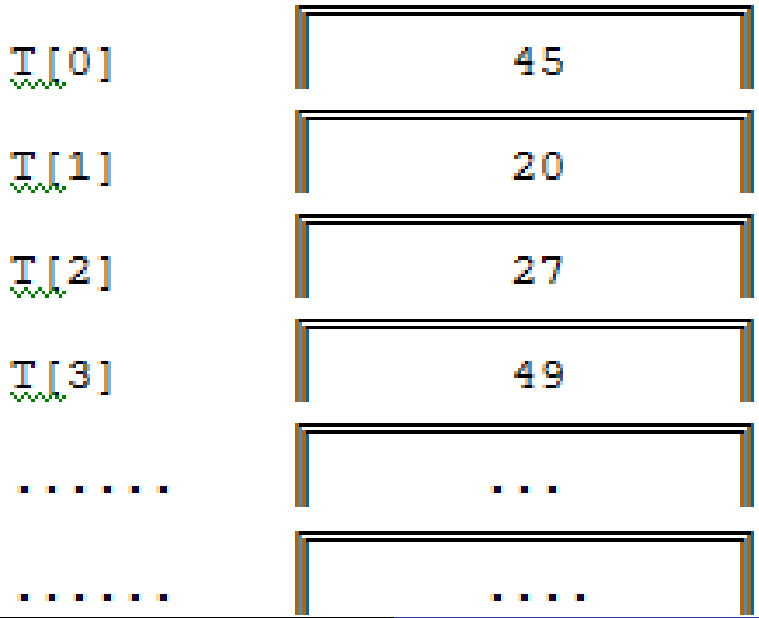
- **Tous** les éléments ont le **même type de base** ;
- Le **nombre** d'éléments stockés est **fini** (fixé) ;

Introduction

Tableau

Un tableau est un ensemble **fini** d'éléments de **même type**, stockés en mémoire à des adresses **contiguës**.

On représente un tableau comme une commode à plusieurs tiroirs dont **chaque tiroir contient un seul objet de même type** :



Introduction

Domaines d'utilisation :

On utilise souvent les **tableaux** pour :

1. **calculer les statistiques de base** (moyenne, écart-type, plus grande et plus petite valeurs , ...)
2. **trier, rechercher** un élément en particulier
3. **afficher** les informations traitées (avant ou après le tri, satisfont aux conditions voulues, etc ...)
4. **calculs matriciels**

Définition

Définitions

- Un **tableau** (uni-dimensionnel) T est une **variable structurée** formée d'un nombre entier N de variables du même type, qui sont appelées les composantes du tableau. **Le nombre de composantes N** est alors la **dimension** du tableau.
- le C permet un accès **direct** et **rapide** aux données d'un tableau.
- En faisant le rapprochement avec les mathématiques, on dit encore que "**T est un vecteur de dimension N**"

Déclaration

Déclaration :

```
<Le_type_des_variables> <Nom_du_tableau>[<dimension
    >];
```

Remarque :

- $<$ *Le_type_des_variables* $>$ est le **type** des élèments du tableau (**int**, **float**, **double**, etc...).
- $<$ *Nom_du_tableau* $>$ est un **identificateur** qui doit respecter les restrictions des identificateurs
- $<$ *dimension* $>$ est **nécessairement une VALEUR NUMERIQUE**. il ne peut être en aucun cas une combinaison des variables du programme.

Déclaration

Façon directe :

```
int age[15] ;

float taille[25], poids[70] ;
```

- **age** est un tableau de 15 entiers. age[0], age[1], ..., age[14] sont 15 éléments du tableau age. Chacun est de type entier (**int**).
- **taille** est un tableau de 25 réels(float). Ses éléments sont taille[0], taille[1], ..., taille[24].
- La façon de déclarer directement les tableaux est simple.

Déclaration

Façon structurée : *#define*

Déclarer les tableaux age, taille, poids, sexe d'un groupe de 150 personnes ou moins.

```
#define MAX_PERS 150

int age[MAX_PERS] ;
float taille[MAX_PERS], poids[MAX_PERS] ;
char sexe[MAX_PERS] ;

int nbPers ; /* le nombre effectif de personnes
    traitées */
```

Déclaration

Façon structurée : const

Utilisation de **const** à la place de *#define*

```
const int MAX_PERS = 150;

float taille[MAX_PERS], poids[MAX_PERS];
    /* non valide en */
char sexe[MAX_PERS]; /* C standard */
etc ...
```

Mémorisation

Mémorisation

- **En C**, le **nom** d'un tableau est le représentant de l'**adresse du premier élément du tableau**. Les adresses des **autres composantes** sont calculées **automatiquement**, relativement à cette adresse.
- Si un tableau possède **N** composantes et si le type déclaré des composantes requiert **M** octets, la mémoire réservée pour ce tableau est de **NxM** octets.
- Exemple :
  `int T[10];`
  Un entier int requiert **2 octets**, il y a 10 éléments, la mémoire réservée pour le tableau T est donc de 2x10=12 octets.

Outline

Initialisation

Initialisation

Il est possible d'**initialiser** un tableau lors de sa **déclaration**, en indiquant la liste des valeurs.
Exemple :

```
int A[5]={100,200,300,400,500};
```

- Le premier élément du tableau, A[0] contiendra la valeur 100.
- le second élément, A[1], contiendra la valeur 200,
- le dernier élément, A[4], contiendra la valeur 500.

Initialisation

Initialisation

Lors de sa **déclaration**, on peut **initialiser le tableau**, **si** le nombre de valeur dans la liste est **inférieur à la dimension** du tableau, les **composantes restantes sont mises à 0**.

```
Exemple:
int A[4]={10,20};
```

- Le premier élément du tableau, A[0] contiendra la valeur 10.
- le second élément, A[1], contiendra la valeur 20,
- le troisième, A[2] sera mis par défaut à 0,
- de même pour le dernier élément, A[3].

Initialisation

Initialisation

Si la **dimension n'est pas indiquée**, l'ordinateur réserve **automatiquement le nombre d'octets nécessaires**.

```
Exemple:
int A[]={10,20,30};
```

Les trois éléments du tableau sont initialisés à 10, 20 et 30, la taille du tableau sera automatiquement mise à 3.

Remarque :

Il faut bien évidemment que le nombre de valeurs dans la liste soit inférieur ou égal à la dimension du tableau !

Initialisation des tableaux

Avant affectation, le contenu du tableau est aléatoire.
Déclaration avec initialisation : type var [taille] = $\{expr1, expr2, ..., exprN\}$

- on doit avoir N< taille,
- si $1 < N < taille$, les initialiseurs manquants sont mis à 0,

Initialisation des tableaux

Exemples corrects

- **int** a[3] ; non initialisé
- int a[3] = {2, 1, 0} ; complètement initialisé
- **int** a[3] = {2, 1} ; équivalent au précédent
- **int** a[ ] = {2, 1, 0} ; équivalent au précédent

Initialisation des tableaux

Exemples incorrects

- **int** a[2] = {2, 1, 3} ; Trop d'inisialisateurs
- **int** a[ ] ; Taille inconnue
- a[2] = {2, 1} ; pas de déclaration

Outline

Accès aux élèments

Accés aux éléèment

On **accède à un élément** du tableau T en lui appliquant l'opérateur **[]**. Les éléments d'un tableau sont toujours numérotés de **0 à nombre-éléments -1**.

Exemples **int** T[15] ;

- T[0] : premier élément,
- T[1] : second élèment, ...,
- T[14] : dernier élèment.

Accès aux élèments

Accés aux élèment

Lorsque l'on déclare un tableau, (par exemple int A[5] ;), on définit un tableau avec **5 composantes**, auxquelles on peut accéder par : A[0], A[1], A[2], A[3], A[4].

Remarque :

- **Le premier** élément du tableau est l'élément **0**,
- donc, pour un tableau de dimension N, le premier sera l'élément 0, le **dernier** l'élément **N-1**.

Outline

Affectation

Avant de pouvoir afficher les composantes d'un tableau, il faut bien sûr leur affecter des valeurs !

Affectation

```
int A[5]={1,2,3,4,5} ;
int i ;
for(i=0 ;i<5 ;i++)
scanf("%d",&A[i]);
```

Remarque :

De la même manière que pour une variable "normale", on indique l'**adresse** (**&**A[i])

Affichage

La structure **for**/**while** se prête particulièrement bien au travail avec les tableaux.

Affichage du contenu d'un tableau

```
int A[5]={1,2,3,4,5};
int i ;
for(i=0 ;i<5 ;i++)
    printf("%3d",A[i]);

Affichage:
1 2 3 4 5
```

Copie d'un tableau

Attention

On ne peut pas affecter un tableau tout entier à un autre.

Ne pas faire

```
int a[10], b[10];
a = b;
```

Copie d'un tableau

Attention

On ne peut pas affecter un tableau tout entier à un autre.

Ne pas faire

```
int a[10], b[10] ;
a = b ;
```

Solution avec boucle

```
int i ;
for (i=0 ;i<10 ;i) a[i] = b[i] ;
```

Tableau multidimensionnelle

Définition et Syntaxe

Tableau multidimensionnel = tableau de tableaux.
matrices,images bitmap, etc.
Syntaxe : type var[taille1][taille2]· · · [tailleN] ; (Pas de limite de dimension)

Exemple : Matrice identité

```
double mat[4][4] ;
int i,j ;
   for (i=0 ;i¡4 ;i++){
     for (j=0 ;j¡4 ;j++){
      if( i== j) mat[i][j] = 1
      else mat[i,j]=0
     }   }
```

Initialisation et copie

- Déclaration avec initialisation :
  `int a[2][3] = { { 1,0,1 }, { 0,1,0 } };`
- Copie de tableaux : par des boucles imbriquées.

Exercices

Exercice 1 :

Ecrire un programme qui déclare et remplisse un tableau contenant les six voyelles de l'alphabet latin.

Exercice 2 :

Ecrire un programme qui déclare un tableau de 9 notes, dont on fait ensuite saisir les valeurs par l'utilisateur.

Exercice 3 :

Ecrire un programme quicalcul la somme et le produit de deux matrice carrées d'ordres (N ,N).

The End