CHAPITRE III LES TABLEAUX & CHAÎNES DE CARACTÈRES

AÏCHA EL GOLLI
aicha.elgolli@essai.ucar.tn

Novembre-décembre 2022

ECOLE SUPÉRIEURE DE LA STATISTIQUE ET DE L'ANALYSE DE L'INFORMATION

DÉFINITION

les tableaux contiennent des données de même type, rangées en mémoire d'une manière contiguë

L'accès aux éléments d'un tableau se fait soit directement par leur numéro d'ordre (**accès direct**), soit en parcourant les cases du tableau dans l'ordre (**accès séquentiel**).

SYNTAXE DE DÉCLARATION

En algorithmique

Nom_tableau [n] : type ;

exemple :

tab[10]:entier;

tab2[100]: réel;

tabc[10]: caractère ;

En C

<type> <nom du tableau> [<nombre d'éléments>] ;

exemple :

int tab[10] ;

double tab2[100] ;

char tabc[10] ;

syntaxe d'accès aux éléments:

<nom du tableau> [<indice>]; // accès à un élément du tableau

LES TABLEAUX

Le type d'un tableau pourra représenter n'importe quel type : simple, structuré ou même vecteur

Il faut prévoir le nombre maximum d'éléments du vecteur afin de réserver la place nécessaire. On dit qu'on a une gestion statique de la mémoire.

LES INDICES

les tableaux en C commence à l'indice 0 obligatoirement pour se terminer à l'indice n-1 (n étant le nombre d'éléments du tableau).

Exemple :

```
int tab1[10] ; …

Printf(" le 1er element du tableau est %d ", tab1[0]) ;

Printf(" le dernier element du tableau est %d ", tab1[9]) ;
```

INITIALISATION DES TABLEAUX

deux cas sont possibles pour l'initialisation de tableau en C :

- ***Soit par une itération en parcourant case par case le tableau et en leurs affectant la valeur d'initialisation ;***
- ***Soit à la déclaration du tableau.***

SYNTAXE D'INITIALISATION

En algorithmique

<nom du tableau>[<nombre d'éléments>] : <type>=(v_1, v_2,…,v_n) ;

Exemple :

tab[3]: entier=(7, 8, 2) ;

tab[ ]: entier=(7, 8, 9) ; {la taille est automatiquement à 3}

En C

<type> <nom du tableau> [<nombre d'éléments>]={v_1, v_2, …, v_n} ;

Exemple :

int tab[3]={7, 8, 2} ;

int tab[ ]={7, 8, 9} ;\\ la taille est automatiquement à 3

EXEMPLE TABLEAU DE CARACTÈRE

Exemple : on déclare un tableau **V** de type **car** et de taille 6 :

V[6] :car =('A', 'B', 'Z', 'A', 'C', 'D') ;{algo}

Char V[6]={'A', 'B', 'Z', 'A', 'C', 'D'};\\en C

1	2	3	4	5	6
A	B	Z	A	C	D

V[2]	'B'
V[-4]	non défini
V[0]	non défini
V[4]	'A'
V[5]	'C'
V[7]	non défini

EXERCICE À FAIRE

Exercice : écrire un algorithme qui permet de saisir les notes des étudiants (notes réelles comprises entre 0 et 20) pour une matière ensuite de les afficher.

Algorithme notes
note[50] : réel ;
nbetudiant, i : entier ;
DEBUT
Afficher (« combien de notes allez vous saisir ? ») ;
Repeter
Entrer (nbetudiant) ;
Jusqu'à (nbetudiant<=50)et(nbetudiant>=0);
POUR i ←0 à nbetudiant-1 **FAIRE**
    Afficher (« donnez la note du », i+1, « ème étudiant ? ») ;
    Entrer (note[i]) ;
FinPOUR ;

POUR i ←0 à nbetudiant-1 **FAIRE**
    Afficher (« La note du », i+1, « ème étudiant est :», note[i]) ;
FinPOUR ;
FIN ;

Exercice : écrire le même algorithme avec des appels de modules

Algorithme notes
note[50] : réel ;
nbetudiant, i : entier ;
DEBUT
Afficher (« combien de notes allez vous saisir ? ») ;
Repeter
Entrer (nbetudiant) ;
Jusqu'à (nbetudiant<=50);
intialise(note, nbetudiant);
affiche(note, nbetudiant);
FIN ;

Procédure initialise(d/r t[]:réel , n:entier)
i:entier;
Début
POUR i ←0 à n-1 **FAIRE**
Afficher (« donnez la note du », i+1, « ème étudiant ? ») ;
Entrer (t[i]) ;
FinPOUR ;
Fin

Procédure affiche(t[]:réel, n:entier)
i:entier;
Début
POUR i ←0 à n-1 **FAIRE**
Afficher (« La note du », i+1, « ème étudiant est :», t[i]) ;
FinPOUR ;
Fin

EXEMPLE : AFFECTATION, COPIE ET AFFICHAGE DE TABLEAUX DE TAILLE DONNÉE

```
#include<stdio.h>
main(){
const int taille=10;
int i;
int A[taille], B[taille];
/* initialisation de A */
for(i=0;i<taille;i++){
printf("Entrer la valeur de la %d eme case du tableau\t", i+1);
scanf("%d",&A[i]);
}
/* Recopie de A en B */
for(i=0;i<taille; i++){
B[i]=A[i];
}
/* Affichage de A */
printf("Les valeurs du tableau A sont: \n");
for(i=0;i<taille; i++){
printf("A[%d]= %d\n",i,A[i]);
}}
```

Interdit d'écrire
B=A

EXERCICES

Exercice 1. Décrire le résultat produit par le programme suivant :

```
#include <stdio.h>
main()
{
int i, b = 0;
int c[I0] = {1, 2, 3, 4, 5 , 6, 7, 8 , 9, 10};
for (i = 0; i < 10; i++){
 if ((c[i] % 2) == 0)
            b = b+ c[i];  }
printf("dans boucle %d %d", i, b);
}
```

EXERCICES

Exercice 2. écrire une procédure en algo qui prend un tableau d'entier et sa taille et échange le contenu de la case d'indice i et le contenu de la case j (les indices sont demandés à l'utilisateur).

Exercice 3. écrire une procédure en algo qui prend un tableau d'entier et sa taille et supprime la case d'indice j (indice demandé à l'utilisateur).

LES TABLEAUX EN C

Exemple de tableau de constantes

const int minmax[2] = { -32768, 32767 }; // ou –32768 et 32767 sont des valeurs constantes.

Si la taille déclarée dépasse le nombre d'éléments :

float tab [7] = { 0.5, 0, -2.90, 0.85, 2.90 } ;

les éléments non-initialisés prennent automatiquement la valeur 0.

tab [0] = 0.5 tab [1] = 0 tab [2] = -2.90 tab [3] = 0.85 tab [4] = 2.30 tab [5] = 0 et tab [6] = 0

LES TABLEAUX DE CARACTÈRE/LES CHAÎNES EN C

Tableau de caractères

char couleur [4] = { 'B', 'L', 'E', 'U' } ; on a

couleur [0] = 'B' couleur [1] = 'L' couleur [2] = 'E' couleur [3] = 'U'

Les chaînes

char **coul** [4] = "BLEU";

char **col** [ ] = "BLEU";

Les tableaux sont différents :

coul [0] = 'B' coul [1] = 'L' coul [2] = 'E' coul [3] = 'U'

col[0] = 'B' col[1] ='L' col[2] ='E' col[3] = 'U' col [4] = '\0'

Dans le tableau "**col**" le compilateur ajoute **automatiquement** le caractère de fin de chaine '**\0**' (correct).

Dans le tableau "**coul**" il n'y a pas de place (donc incorrect). Déclaration correcte : char coul [5] ="BLEU".

TABLEAU MULTIDIMENSIONNEL

Un tableau multidimensionnel est un tableau composé d'autres tableaux. On peut se représenter facilement un tableau à deux dimensions ; c'est un tableau qui regroupe des tableaux. Mais un tableau tri- voir quadridimensionnel est beaucoup plus difficile à concevoir dans notre esprit. Ce serait un tableau de tableaux eux-mêmes composés de tableaux à leur tour composés de tableaux… Mais un tableau multidimensionnel en mémoire est représenté comme un vecteur : de façon linéaire. Voici comment on déclare un tableau bidimensionnel de 100 éléments :

```
int tableau[2][50];

// tableau[2][50] :entier ; {en algorithmique}
```

Ceci équivaut à un tableau unidimensionnel de 100 éléments, seul l'indexage change. Le tableau possède 2 lignes et 50 colonnes.

TABLEAU MULTIDIMENSIONNEL

Les éléments du tableau sont eux-mêmes des tableaux.
Déclaration :
TYPE NOM [TAILLE1] [TAILLE2] …. [TAILLEN]; **(en c)**
NOM [TAILLE1] [TAILLE2] …. [TAILLEN] : TYPE; **(en Algo)**

TAILLEi = nombre de composantes associées à la dimension i.
Exemple : Tableau à 2 dimensions (matrice).
Déclaration : int tab[3][4] ;//3 lignes et 4 colonnes.
Initialisation dans le programme :
int mat [3][4] = { {5, 3, -2, 42}, {44, 15, 0, 6}, {97, 6, 81, -21} }
L'élément de valeur –21 est désigné par : mat [2][3] (3ème ligne, 4ème colonne)

Col0	Col1	Col2	Col3	
5	3	-2	42	Ligne0
44	15	0	6	Ligne1
97	6	81	-21	Ligne2

TABLEAU MULTIDIMENSIONNEL

On peut initialiser un tableau multidimensionnel de cette façon :

```
int tableau [20][40], i, j;

for (i=0; i < 20; i++){

    for (j=0; j<40; j++)

        tableau[i][j] = i*40+j;

}
```

MANIPULATION DES TABLEAUX ET DES CHAÎNES

Une chaîne de caractères, ce n'est rien d'autre qu'un tableau de caractères. Ainsi, on peut définir la variable chaîne comme cela :

char chaine[] = "petit texte";

Note : toute chaîne de caractères se termine par le caractère nul '\0'. Ainsi, chaine[0] désignera par exemple le caractère 'p', chaine[1] le caractère 'e', etc. Mais que désigne chaine ? Il s'agit de l'adresse, dans la mémoire de l'ordinateur, à laquelle est stocké le premier élément du tableau. On appelle cela un pointeur sur chaine[0]. L'instruction 'char phrase[80];' déclare une chaîne qui ne pourra contenir que 79 caractères !! Le premier caractère sera phrase[0], le dernier phrase[78]. La grande majorité des opérations effectuées sur des chaînes nécessitent d'inclure la bibliothèque <string.h>.

Une chaîne peut être représentée soit par **un tableau** soit avec **un pointeur**.

MÉTHODES POUR ENTRER UNE CHAÎNE DANS UN TABLEAU

a) Déclaration et initialisation du tableau avec la chaîne :

Exemple : char texte [ ] = "ceci est un exemple ";

b) Utilisation de la fonction strcpy (de STRING COPY) :

Exemple : char tab [30]; //déclaration d'un tableau de 30 caractères

strcpy (tab, "ceci est un exemple");

c) Utilisation de la fonction gets (de GET STRING):

Exemple : char tab [30];

printf ("Entrez la chaîne ");

gets (tab); //chaîne saisie à l'écran

MÉTHODES POUR ENTRER UNE CHAÎNE DANS UN TABLEAU

d) scanf peut lire des chaînes mais s'arrête au premier blanc :
Exemple : scanf ("%s ", tab); // ne lit que la chaîne "ceci"

e) scanf peut lire la chaîne caractère par caractère

```
#include <stdio.h>
main (){
int i;
char tab [ 30 ];
printf ("Entrez la chaîne "); // on entre "ceci est un exemple"
for (i = 0; i < 10; ++ i)  // lecture de la chaîne caractère par caractère
 scanf ("%c", &tab[ i ]); // &tab[ i ] désigne l'adresse du caractère i
}
```

MÉTHODES POUR ENTRER UNE CHAÎNE DANS UN TABLEAU

L'initialisation d'une variable chaîne de caractère à la déclaration se fait donc ainsi :

1) char chaine1[10]={'s','a','l','a','h','\0'} ;
2) char chaine2[]={'s','a','l','a','h','\0'} ; //6 caractères
3) char chaine3[10]= "salah" ;
4) char chaine4[]= "salah" ;

AFFICHAGE DES CHAÎNES AVEC PUTS (DE PUT STRING)

```
#include <stdio.h>
main (){
char tab [ ] = "ceci est un exemple";
puts ( tab ); //affiche : ceci est un exemple
}

#include <stdio.h>
#include <string.h>
main (){
char tab [ 30 ];
strcpy ( tab, "ceci est un exemple");
puts ( tab ); //affiche la même chose.}
```

COPIE DE CHAÎNES

```
#include <stdio.h>
void main (){
char src [ ] = "bonjour !"; char dest [ ] = "salut les copains";
int i = 0;
while ( src [ i ] != ' \0' )//ou != 0 {
            dest [ i ] = src [ i ];
i ++;}
puts ( dest );}
//affichage ? ➔ bonjour ! copains

while ( (dest [ i ] = src [ i ]) != ' \0' )
i ++;
puts ( dest );} // affichage? ➔ bonjour !
```

LES FONCTIONS DE <STRING>

La bibliothèque <*string*> fournit une multitude de fonctions pratiques pour le traitement de chaînes de caractères. Voici une brève description des fonctions les plus fréquemment utilisées.

Dans le tableau suivant, <n> représente un nombre du type **int**. Les symboles <s> et <t> peuvent être remplacés par :

* une chaîne de caractères constante

* le nom d'une variable déclarée comme tableau de **char**

* un pointeur sur **char** (à découvrir plus tard dans le chapitre portant sur les pointeurs)

FONCTIONS POUR LE TRAITEMENT DE CHAÎNES DE CARACTÈRES

Dans le tableau suivant, <n> représente un nombre du type **int.**

Les symboles <s> et <t> peuvent être remplacés par :

* une chaîne de caractères constante

* le nom d'une variable déclarée comme tableau de **char**

* un pointeur sur **char** (à découvrir plus tard dans le chapitre portant sur les pointeurs)

Fonction	Descriptions
strlen(<s>)	fournit la longueur de la chaîne sans compter le '\0' final
strcpy(<s>, <t>)	copie <t> vers <s>
strcat(<s>, <t>)	ajoute <t> à la fin de <s>
strcmp(<s>, <t>)	compare <s> et <t> lexicographiquement et fournit un résultat: négatif : si <s> précède <t> zéro : si <s> est égal à<t> positif : si <s> suit <t>
strncpy(<s>, <t>, <n>)	copie au plus <n> caractères de <t> vers <s>
strncat(<s>, <t>, <n>)	ajoute au plus <n> caractères de <t> à la fin de <s>

REMARQUES

- Comme le nom d'une chaîne de caractères représente une adresse fixe en mémoire, on ne peut pas 'affecter' une autre chaîne au nom d'un tableau:

Il faut bien copier la chaîne caractère par caractère ou utiliser la fonction strcpy respectivement strncpy:

strcpy(A, "Hello");

- La concaténation de chaînes de caractères en C ne se fait pas par le symbole '+' comme en langage algorithmique ou en Pascal. Il faut ou bien copier la deuxième chaîne caractère par caractère ou bien utiliser la fonction strcat ou strncat.