Programmation en C/C++

Série 3c
Les tableaux


Important !

Difficile !

Objectifs

Utiliser et comprendre la structure de données de type tableau. Utilisation conjointe des tableaux et des boucles. Premiers algorithmes.

Introduction

On utilisera une structure de données de type tableau lorsque plusieurs données **du même type** doivent être manipulées. En effet, au lieu d'utiliser de nombreuses variables différentes, un simple tableau où chaque donnée est repérée par son indice permettra de manipuler facilement celles-ci, à l'aide de boucles (structure répétitive).

Déclaration, initialisation, affectation

Un tableau se déclare avec l'opérateur *[ ]*

Exemple : déclaration d'un tableau de 20 caractères se nommant T
char T [20];

Il est possible d'initialiser un tableau lors de sa déclaration en mettant les valeurs du tableau entre accolades :

Exemple : déclaration et initialisation d'un tableau de 5 entiers se nommant Tab
int Tab [5] = {3, 2, 8, -2, -30};

Exemple : déclaration et initialisation d'un tableau de 6 caractères se nommant T1
char T1 [] = {'A', 'B', 'C', ';', 'O', 'Z'};

Rq : on n'est pas obligé de préciser sa taille, qui est implicite dans ce cas.

Affectation

Tab[0] = 1; J'affecte 1 à la première case du tableau Tab (indice 0)

Notion d'indice

Le premier élément d'un tableau *Tab* de n éléments a pour premier indice *0* et *n-1 pour* dernier indice.

Exemple

int Tab [5] = {3, 2, 8, -2, -30};
printf("%d", Tab[0]); : Affiche le 1er élément de *Tab*, c'est-à-dire 3
printf("%d", Tab[3]); : Affiche le 4ème élément de *Tab*, c'est-à-dire -2
Attention, ici, le dernier élément de *Tab* est *Tab[4]*

Attention ! Ce tableau a 5 éléments mais Tab[5] n'existe pas !
On parle de débordement lorsqu'on cherche à atteindre une case d'un tableau alors qu'elle n'existe pas.

Exemple Initialisation de toutes les cases d'un tableau à 10 :

```
int Tab[8], i;
for(i=0; i<8 ;i++)
        Tab[i] = 10;
```

Taille d'un tableau

La taille d'un tableau, en octets, est donné par l'instruction *sizeof().*
Exemple : Pour un tableau *T* d'entiers : *printf("%d" ; sizeof(T));*

Nombre d'éléments d'un tableau

Le nombre d'éléments d'un tableau est donné par la division de *sizeof (tableau) par sizeof (type des variables du tableau). En fait, o*n divise la taille du tableau par la taille d'un élément du tableau.
Exemple : pour un tableau d'entiers nommé *T* : *sizeof(T)/sizeof(int)*

On pourra s'en servir dans une boucle :

```
int Tab [12];
for (i=0 ; i <  sizeof(Tab)/sizeof(int) ; i++)
        …..;
```

Tableaux à deux dimensions

Un tableau peut posséder 2 dimensions : il possèdera alors plusieurs lignes et colonnes. On aura une déclaration du type *int T [i] [j];* où *i* correspond au nombre de lignes et *j* au nombre de colonnes du tableau.
Attention ! La première ligne a pour indice *i =0*, la première colonne a pour indice *j=0*.

Exemple

4	12	6	76	90	819
3	15	567	39	238	375

Le tableau précédent pourra être déclaré comme suit :
int T [2] [6]; le premier nombre correspond au nombre de lignes : il y en a 2, le second au nombre de colonnes : il y en a 6.

On pourra affecter des valeurs à ce tableau comme suit :

```
T = {   {4,12,6,76,90,819},
        {3, 15,567,39,238,375}
};
```

Pour saisir ou afficher les valeurs des différentes lignes et colonnes du tableau il faudra jouer sur les indices *i* et *j*. *T [0] [3]* vaut 76 dans notre exemple. Pour accéder aux données on utilisera la structure de boucle ; si on souhaite balayer tout le tableau il faudra utiliser des boucles imbriquées.

Déclaration dynamique d'un tableau

On peut ne pas connaître le nombre de cellules d'un tableau au moment de sa déclaration.
Déclarer la taille du tableau dans le code source peut poser 2 types de problèmes :

Insuffisance de mémoire
On a déclaré un tableau de 15 entiers et on doit finalement en stocker 20 : on est confronté ici à un problème important : on ne pourra pas stocker les 5 derniers entiers, faute d'une mémoire insuffisante.

Gaspillage de mémoire
Exemple, pour une course cycliste : on a déclaré un tableau de 100 entiers qui correspondront aux numéros de dossards de 100 coureurs inscrits (maximum) mais, finalement, 45 coureurs s'inscrivent : on gaspille de la mémoire pour rien. Cela peut être problématique pour des contextes "informatique embarquée" pour des terminaux mobiles ne possédant que peu de mémoire.

Heureusement, le langage C a prévu ce cas de figure : il sera possible de définir la taille du tableau un peu plus tard lors de l'exécution du programme.
On parlera d'allocation dynamique de mémoire. On ne traitera ici le cas qu'à travers un exemple pour y revenir plus en détail lors d'un prochain cours sur les pointeurs.

Il faudra utiliser la librairie C nommée *stdlib.h* (*#include <stdlib.h>),* les mots clés *malloc* (allocation dynamique de mémoire) et *free* (libération de la mémoire), *sizeof()* (taille d'une variable).

Remarque
Sizeof (int) renverra la valeur 4 car un entier est codé sur 4 octets.

Exemple
On veut créer dynamiquement un tableau (MonTab) d'entiers.

1- On va déclarer une variable nommée *Taille* de type entier qui stockera le nombre d'entiers présents dans le tableau : *int Taille;*
2- On va créer un pointeur sur le tableau d'entiers : *int* MonTab;*
3- On demande de saisir la valeur de *Taille* : scanf("%d", &Taille);
4- On alloue dynamiquement de la mémoire pour le tableau *MonTab* qui possèdera *Taille* entiers : Mon*Tab = malloc(Taille * sizeof(int));*
5- On réalise la saisie des données comme d'habitude…
6- On libère la mémoire quand on a fini d'utiliser le tableau : *free (MonTab);*

```
//Déclaration du pointeur sur le tableau d'entiers
int* MonTab;

// saisie de la taille du tableau
scanf("%d", &Taille);

//allocation dynamique de mémoire
//on multiplie la taille d'un entier par le nombre d'entiers (Taille)
MonTab =  malloc(Taille * sizeof(int));

//Saisie… comme d'habitude
for (i=0; i<Taille; i++) {
 printf("Saisie %d : \t ",(i+1));
scanf("%d", &MonTab[i]);
}

//libération de la mémoire en fin de programme
free (Tab);
```

EXERCICES

Exercice 1

Énoncé

Écrire le programme qui affecte (sans saisie au clavier) les voyelles (a, e, i, o, u, y) dans un tableau et qui les affiche ensuite.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int i;
char T1 [] = {'a', 'e', 'i', 'o', 'u', 'y'};

for ( i=0; i<6; i++)
        printf("%c\t",T1[i]);

return 0;
}
```

Exercice 2

Énoncé

Écrire le programme qui affecte (sans saisie au clavier) les multiples de 5 compris entre 0 et 50 dans un tableau et qui les affiche ensuite à l'aide d'une seconde boucle.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int i;
int T[11];

// Boucle d'affectation des valeurs
for ( i=0; i<=10; i++)
        T[i]=5*i;

//Boucle d'affichage
for (i=0; i<=10; i++)
        printf("%d\t",T[i]);

return 0;
}
```

Exercice 3

Énoncé

Écrire le programme qui affecte (sans saisie au clavier) les voyelles (a, e, i, o, u, y) dans un tableau. Le programme demande ensuite de saisir un indice du tableau (entre 0 et 5) et affiche alors la voyelle située à cet indice du tableau.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int i;
int indice;

char T1 [] = {'a', 'e', 'i', 'o', 'u', 'y'};

for ( i=0; i<sizeof(T1)/sizeof(char); i++)
        printf("%c\t",T1[i]);

printf("\nSaisissez votre indice");
scanf("%d",&indice);
printf(" Resultat : %c\n",T1[indice]);

return 0;
}
```

Exercice 4

Énoncé

Écrire le programme qui affecte (sans saisie au clavier) les nombres *1,13, 45, 57, 234, 450* dans un tableau. Le programme demande ensuite de saisir une valeur et affiche ensuite son rang (position) dans le tableau.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int i, indice=0;
int valeur;
int T1 [] = {1,13,45,57,234,450};

for ( i=0; i<sizeof(T1)/sizeof(int); i++)
        printf("%d\t",T1[i]);

printf("\nSaisissez votre valeur\t");
scanf("%d",&valeur);
```

```
for (i=0; i< sizeof(T1)/sizeof(int); i++) {
        if (T1[i]==valeur)      {
                indice = i;
                break;
        }
}

printf("\nRang de votre valeur : %d\n",++indice);
return 0;
}
```

Rq : Exceptionnellement, on a choisi de placer un break dans la boucle : en effet, une fois le nombre trouvé, il n'est pas nécessaire de continuer à balayer le reste du tableau : le *break* permet de sortir de la boucle.

Exercice 5

Énoncé

Écrire un programme qui demande de saisir 6 nombres réels au clavier et les affiche ensuite, à l'aide d'une seconde boucle.

Corrigé

```
#include <stdio.h>
int main()
{
float T[6];
int i;

for (i=0; i<sizeof(T)/sizeof(float); i++) {
      printf("Saisissez la valeur %d : \t ",(i+1));
      scanf("%f", &T[i]);
}

for (i=0; i< sizeof(T)/sizeof(float); i++)
      printf("Valeur %d : %.2f\n",(i+1), T[i] );

return 0;
}
```


Exercice 6

Énoncé

Écrire un programme qui demande de saisir 8 notes (réels) au clavier et affiche la moyenne de ces notes avec 2 chiffres après la virgule, ensuite.

Corrigé

```
#include <stdio.h>
```

```
int main()
{
 float notes[8], somme = 0;
   int i;

   for (i=0; i< sizeof(notes)/sizeof(float); i++) {
      printf("Saisissez la note %d\n ",(i+1));
      scanf("%f", &notes[i]);
      somme += notes[i];
   }
   printf("La moyenne vaut :  %.2f\n", (somme/ sizeof(notes)/sizeof(float)) );
   return 0;
}
```

Remarque : Dans ce programme, *sizeof(notes)/sizeof(float)* vaut 8.
Pour afficher la moyenne on divise, après être sorti de la boucle, la valeur contenue dans la variable *somme* par *sizeof(notes)/sizeof(float).*

Exercice 7

Énoncé

Écrire le programme qui demande de saisir 5 réels au clavier, puis les affiche ainsi que le plus grand élément de ce tableau.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int a;
float T [5];
float max;

for ( a=0; a< sizeof(T)/sizeof(float); a++)   {
        printf("Saisir un reel:");
        scanf("%f",&T[a]);
}

for(a=0;a< sizeof(T)/sizeof(float);a++)
        printf("%.2f\n",T[a]);

max =T[0];
for(a=0; a< sizeof(T)/sizeof(float); a++)          {
        if(T[a]>max)
                max = T[a];
}
printf("Le plus grand element est %.2f\n", max);

return 0;
}
```

Exercice 8

Énoncé

Écrire le programme qui demande de saisir 5 réels au clavier, puis les affiche ainsi que le plus petit élément de ce tableau.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int a;
float T [5];
float min;

for ( a=0; a<5; a++)   {
        printf("Saisir un reel:");
        scanf("%f",&T[a]);
}
for(a=0;a<5;a++)
        printf("%.2f\n",T[a]);

min =T[0];
for(a=0; a<5; a++)                {
        if (T[a]<min)
                min = T[a];
 }
printf("Le plus petit element est %.2f\n", min);

return 0;
}
```

Exercice 9

Énoncé

Écrire le programme qui demande de saisir 5 réels au clavier, puis les affiche ainsi que le plus petit élément de ce tableau et le rang de ce dernier au sein du tableau.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int a, IndiceMin =0;
float T [5];
float min;
```

```
for ( a=0; a< sizeof(T)/sizeof(float); a++)   {
        printf("Saisir un reel:");
        scanf("%f",&T[a]);
}
for(a=0;a< sizeof(T)/sizeof(float);a++)
        printf("%.2f\n",T[a]);

min =T[0];
for(a=0; a< sizeof(T)/sizeof(float); a++)  {
        if (T[a]<min)          {
                min = T[a];
                IndiceMin = a;
                }
 }
printf("Le plus petit element est %.2f; son rang : %d\n ", min,(++ IndiceMin));

return 0;
}
```

Rq : Attention! On incrémente *IndiceMin* avant de l'afficher.

Exercice 10

Énoncé

Soit le tableau suivant :

1	100	200	300	400

Écrire le programme qui enregistre les valeurs de ce tableau, sans saisie au clavier, puis affiche les valeurs dans l'ordre inversé sans utiliser d'autre tableau.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>
int main() {
int i ;

int T1 [] = {1,100,200,300,400};

for ( i=4; i>=0; i--)
        printf("%d\t",T1[i]);

return 0;
}
```

Exercice 11

Énoncé

Écrire le programme qui demande de saisir 5 réels au clavier dans un premier tableau *T*, puis les recopie de façon inversée dans un second tableau *T2*. Le programme affichera les éléments du premier tableau puis ceux du second, comme montré dans l'exemple ci-après.

Exemple :

```
saisir l'entier numero  1 :
2
saisir l'entier numero  2 :
4
saisir l'entier numero  3 :
6
saisir l'entier numero  4 :
8
saisir l'entier numero  5 :
10

Affichage tableau 1
2       4       6       8       10
Affichage tableau 2
10      8       6       4       2
```

Corrigé

```
#include<stdio.h>
#include<stdlib.h>
main() {

int i;
int T[5];
int T2[5];

for(i=0; i< sizeof(T)/sizeof(int); i++) {
        prinft( "saisir l'entier de la case %d:\n",(i+1));
        scanf("%d",&T[i]);
        }

//affichage
for(i=0; i< sizeof(T)/sizeof(int); i++)
    printf( "%d", T[i]);

//recopie inversée
for (i=0; i< sizeof(T)/sizeof(int); i++)
        T2[4-i]=T[i];

//affichage
for (i=0;i< sizeof(T)/sizeof(int);i++)
        printf("%d",T2[i]);

return 0;
```

```
}
```

Exercice 12

Énoncé

Soit le tableau suivant :

1	100	200	300	400

Écrire le programme qui inverse les valeurs de ce tableau, sans saisie au clavier, dans ce même tableau. Consigne : n'utiliser qu'un seul tableau !
A l'arrivée on aura :

400	300	200	100	1

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int i, tampon;
int T1 [] = {1,100,200,300,400};
tampon = T1[0];

for ( i=0; i<((sizeof(T1)/sizeof(int))/2); i++) {
        tampon = T1[i];
        T1[i]=T1[4-i];
        T1[4-i]=tampon;
        }

for ( i=0; i<((sizeof(T1)/sizeof(int))/2); i++)
        printf("%d\t",T1[i]);

return 0;
}
```

Rq

Le premier élément est inversé avec le dernier, le deuxième avec l'avant dernier et ainsi de suite… Il faut s'arrêter à la moitié du tableau car autrement on inverse une deuxième fois et on retombe sur le tableau de départ!

Exercice 13

Énoncé

Soit le tableau à deux dimensions suivant :

1	2
11	22
111	222

Écrire le programme qui enregistre les valeurs de ce tableau, sans saisie au clavier, puis affiche chacun des prix des différents articles.

```
1       2
11      22
111     222
```

Capture d'écran de l'affichage

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int i;
int T[3][2] = {
        {1,   2  },
        {11,  22 },
        {111, 222}
};

for ( i=0; i<3; i++)
        printf("%d\t %d\n",T[i][0], T[i][1]);
return 0;
}
```

Exercice 14

Énoncé

Soit le tableau à deux dimensions suivant :

Numéro	1234	1256	4578
Age	12	20	30

Écrire le programme qui gère les nombres de ce tableau et affiche, sur chaque ligne, le numéro d'une personne et son âge.

```
Numero   Age
1234     12
1256     20
4578     30
```

Capture d'écran de l'affichage

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int j;
int T[2][3] = {
        {1234,   1256, 4578 },
        {12,20 ,30 }
};
```

```
printf("Numero\t Age\n");

for ( j=0; j<3; j++)
        printf("%d\t %d\n",T[0][j], T[1][j]);

return 0;
}
```

Exercice 15

Énoncé

Soit le tableau à deux dimensions suivant :

Référence	1234	1256	4578
Age	12	20	30

Écrire le programme qui gère les nombres de ce tableau et affiche la moyenne des âges des personnes.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>
int main() {
float somme;
int j;
int T[2][3] = {
{1234,   1256, 4578 },
{12,20 ,30 }
};
for ( j=0; j<3; j++)
        somme +=T[1][j];

somme/=3.0;
printf("Moyenne : %.2f\n",somme);

return 0;
}
```

Exercice 16

Énoncé

Écrire le programme qui transforme le tableau suivant :

1234	1256	4578
12	20	30

En :

12	20	30
1234	1256	4578

Et l'affiche en colonnes avant et après modification.
On n'utilisera qu'un seul tableau à 2 dimensions !

```
1234    12
1256    20
4578    30

12      1234
20      1256
30      4578
```

Affichage obtenu

Corrigé

```
#include<stdio.h>
#include<stdlib.h>

int main() {

float somme;
int j;
int tampon;
int T[2][3] = {
                {1234,   1256, 4578 },
                {12,20 ,30 }
};

// affichage avant inversion
for ( j=0; j<3; j++)
        printf("%d\t %d\n",T[0][j], T[1][j]);

for ( j=0; j<3; j++) {
        tampon = T[0][j];
        T[0][j]=T[1][j];
        T[1][j]=tampon;
};

// affichage après inversion
printf("\n");
for ( j=0; j<3; j++)
        printf("%d\t %d\n",T[0][j], T[1][j]);

return 0;
}
```

Exercice 17 Allocation dynamique de mémoire

Énoncé

Créer un programme qui va créer un tableau de réels (*float*), demande leur saisie au clavier puis les affiche à l'écran. Contrainte : l'allocation de mémoire pour le tableau sera dynamique : on demande le nombre de cases du tableau puis on réserve la mémoire nécessaire pour réaliser ensuite la saisie et le stockage des données (utiliser *malloc*, *sizeof()* et *free* vus en cours).

```
Saisissez le nombre de cases du tableau :        5
Saisissez la valeur 1 :          2.44
Saisissez la valeur 2 :          3.44467
Saisissez la valeur 3 :          456
Saisissez la valeur 4 :          700.00098
Saisissez la valeur 5 :          0
Valeur 1 : 2.44
Valeur 2 : 3.44
Valeur 3 : 456.00
Valeur 4 : 700.00
Valeur 5 : 0.00
```

Corrigé

```
#include <stdio.h>
#include <stdlib.h>

int main() {
int taille, i;
float* Tab;

printf("\nSaisissez le nombre de cases du tableau : \t ");
scanf("%d", &taille);

//allocation dynamique de mémoire
Tab =  malloc(taille * sizeof(float));

//Saisie
for (i=0; i<taille; i++) {
 printf("Saisissez la valeur %d : \t ",(i+1));
scanf("%f", &Tab[i]);
}

//affichage
for (i=0; i<taille; i++)
        printf("Valeur %d : %.2f\n",(i+1), Tab[i] );

//libération de la mémoire
free (Tab);

return 0;
```

```
}
```