|  |  |  |
| --- | --- | --- |
|  | **INSTITUT SUPERIEUR DES ETUDES TECHNOLOGIQUES DE NABEUL** | |
|  | Travaux Pratiques N° 4 : les tableaux | |
| **Classes : TI 1** | | **Enseignante : Sana REFAI** |

## 1. Objectifs

Le but de ce TP est de se familiariser avec les tableaux unidimensionnel et bidimensionnel en C.

## 2. Définition

Un tableau est une structure de donnée permettant de stocker des données de même type. Chacune des valeurs est repérée par un indice indiquant la position de la donnée dans le tableau.

Un tableau est caractérisé par :

* Son nom;
* Sa taille;
* Sa dimension;
* Le type de ses éléments

## 3. Tableau à une dimension (uni-dimensionnel)

Un tableau uni-dimensionnel (vecteur) est une manière de ranger des éléments ou des valeurs de même type. Il regroupe ces éléments dans une structure fixe et permet d’accéder à chaque élément par l’intermédiaire de son rang ou indice.

Tableau A

..…

**N composantes**

**A :** Tableau uni-dimensionnel de taille N.

### 3.1 Déclaration

En langage C, la déclaration d’un tableau à une dimension est définie comme suit :

|  |
| --- |
| **Syntaxe** |
| **<TypeSimple> <NomTableau> [<Taille>];** |
| **Exemple** |
| **float Notes [20];**  Exemple 1 :  **int A [100];**  **char Alphabet [26];**   * <Taille> est nécessairement une valeur numérique. Ce ne peut être en aucun cas une combinaison des variables du programme. * Pour un tableau de taille N, le compilateur C réserve N places en mémoire pour ranger les éléments du tableau. * Un élément du tableau est repéré par son indice. En langage C les tableaux commencent à l'indice 0. L'indice maximum est donc N-1. * En langage C, le nom d’un tableau est le représentant de l’adresse du premier élément du tableau.   Exemple 2  **short A[5] = {1200, 2300, 3400, 4500, 5600};**   * Les adresses des autres composantes sont calculées (automatiquement) relativement à cette adresse. * Espace occupé en mémoire : Taille d’une composante en octets x N   Exemple 3  **short A [4]; /\* Taille = 2 octets x 4 \*/**  **int TAB [10]; /\* Taille = 4 octets x 10 \*/** |

### 3.2 Initialisation

Lors de la déclaration d’un tableau, on peut initialiser les composantes du tableau en indiquant la liste des valeurs respectivement entre {}.

Exemple:

* **s**
* **hort A[5] = {1, 10, 9, 4, 5};**

**Réservation de (2 x 5) octets.**

* + **char Lettres [26] = {‘ A ’, ‘ B ’};**

**Les autres composantes initialisées à 0.**

* + **float B [ ] = {2.4, 1.6, 3.33};**

**Réservation automatique de (4x3) octets (la dimension n’est pas indiquée explicitement).**

* + **short A[4] = {1, 10, 9, 4, 5};**

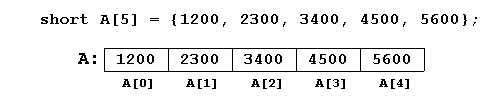
**Erreur!**

### 3.3 Accès

**<NomTableau> [<indice>]**

Pour un tableau T de taille N:

* T
* [0] pour accéder au premier élément
* T[N-1] pour accéder au dernier élément



### 3.4 Affectation

Syntaxe:

**<NomTableau>[<indice>] = <expression>;**

**T[2]=3;**

Exemple:

**int Notes [100] , i;**

**for (i = 0; i < 100; i++) /\* Remplissage du tableau \*/**

**{**

**printf (“Entrez la note de l ’étudiant N° %d\n“, i+1 );**

**scanf (“%d “, &Notes[i]);**

**}**

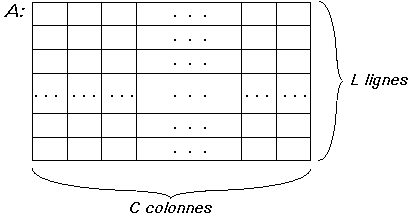
## 4. Tableau Multidimensionnel

Les tableaux multidimensionnels sont des extensions des tableaux à un seul indice, ils servent à représenter des objets plus complexes comme par exemple les matrices.

Un tableau à N dimensions est en fait un tableau unidimensionnel de tableaux de N-1 dimensions.

En effet, un tableau A à 2 dimensions est à interpréter comme un tableau (uni-dimensionnel) de taille L dont chaque composante est un tableau (uni-dimensionnel) de taille C. Donc on a :

* L : Nombre de lignes
* C : Nombre de colonnes
* L x C : Nombre de composantes



### 4.1 Déclaration

En langage C, la déclaration d’un tableau à deux dimensions est définie comme suit :

Syntaxe:

**<TypeSimple> <NomTabl> [<TailleLigne>][<TailleCol>];**

Exemple:

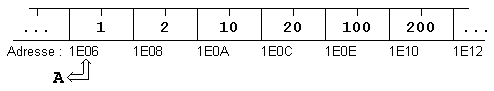
**int Notes [3][20]; /\* 20 étudiants, 3 matières \*/**

**char T[12][18];**

**float A[2][2];**

* Le nom d’un tableau est le représentant de l’adresse du premier élément du tableau
* Les composantes d’un tableau à 2 dimensions sont stockés ligne par ligne dans la mémoire
* Espace occupé en mémoire: Taille d’une composante en octets x L x C

**short A[3][2]={{1,2},{10,20},{100,200}};/\*Taille = 2 octets x3x2\*/**



### 4.2 Initialisation

Lors de la déclaration d’un tableau, on peut initialiser les composantes du tableau en indiquant la liste des valeurs respectivement entre {}. A l’intérieur de la liste, les composantes de chaque ligne sont entre {}.

Exemple:

**short A[3][5] = {{1,10,9,4,5}, {1,1,3,4,5}, {1,10,3,4,4}};**

**Réservation de (2 x 3 x 5) octets**

**int C [4][4] = {{1, 1, 1, 1}};**

**Les autres composantes initialisées à 0**

**float B [ ][3] = { {2.4, 1.6, 3.33},{0.5, 0.2, 0.1}} ;**

**Réservation automatique de (4x2x3) octets Pour les lignes seulement!**

**int C [4][4] = {{1, 1, 1, 1, 1}}; Erreur!**

### 4.3 Accès

**<NomTableau> [<Ligne>][<Colonne>]**

Les éléments d'un tableau de dimensions L et C se présentent de la façon suivante:

**/ \**

Syntaxe:

**| A[0][0] A[0][1] A[0][2] . . . A[0][C-1] |**

**| A[1][0] A[1][1] A[1][2] . . . A[1][C-1] |**

**| A[2][0] A[2][1] A[2][2] . . . A[2][C-1] |**

**| . . . . . . . . . . . . . . . |**

**| A[L-1][0] A[L-1][1] A[L-1][2] . . . A[L-1][C-1] |**

**\ /**

Exemple:

Pour un tableau T de taille L et C :

* T[0][0] pour accéder au premier élément
* T[L-1][C-1][ pour accéder au dernier élément

### 4.4 Affectation

Syntaxe:

**<NomTableau> [<Ligne>][<Colonne>] = <expression>;**

Exemple:

**T[0][1] = 2,**

**int Notes [5][100];**

**int i, j;**

**for (i = 0; i < 5; i++)**

**{**

**printf (“Matière %d \n“, i+1);**

**for (j=0; j < 100; j++)**

**{**

**printf (“Entrez la note de l ’étudiant n° %d\n“, j+1);**

**scanf (“%d “, &Notes[i][j]);**

**}**

**}**

## 5. Les Exercices

### 5.1 Exercice 1

On a enregistré les températures maximales et minimales de la région de Nabeul pendant 12 mois de l’année 2020. On aimerait connaitre la température moyenne pour chaque mois.

Ecrire un programme C qui saisit au clavier les deux tableaux Tmax et Tmin puis remplit le tableau Tmoy à partir des 2 tableaux précèdent..

Afficher le tableau Tmoy.

### 5.2 Exercice 2

Etant donné un entier positif X, on se propose d’écrire un algorithme qui permet de donner son code binaire suivant cette méthode:

1. On divise (division entière) le nombre X par 2.

2. On sauvegarde le reste de la division.

3. On refait les deux étapes précédentes, jusqu’à avoir un quotient nul.

4. Le regroupement des restes en sens inverse de leurs apparitions donne la valeur du nombre X en binaire.

Exemple:

Si X=13 alors

 la division entière de 13 par 2 donne un quotient =6 et un reste = 1

 la division entière de 6 par 2 donne un quotient =3 et un reste = 0

 la division entière de 3 par 2 donne un quotient =1 et un reste = 1

 la division entière de 1 par 2 donne un quotient =0 et un reste = 1

Donc le nombre décimal 13 vaut 1101 en binaire.

### 5.3 Exercice 3

Ecrire un algorithme qui permet de :

* Saisir un tableau T de N entiers strictement positifs et inferieur ou égale a 10 ( 2 ≤ N ≤ 100 )
* Créer à partir de T, un tableau T\_occurrences contenant le nombre d’occurrences de chaque entier dans T. Calculer la taille de T\_occurences.

Exemple :

N  : 11

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| T : | 10 | 6 | 8 | 2 | 8 | 2 | 6 | 6 | 2 | 1 | 6 |
|  |  |  |  |  |  |  |  |  |  |  |  |
| T\_occurences : | 1 | 3 | 0 | 0 | 0 | 4 | 0 | 2 | 0 | 1 |  |
| L’entier 2 apparaît 3 fois | 1 | 2 | 3 | 4  L’entier 6 apparaît 4 fois | 5 | 6 | 7  L’entier 9 apparaît 0 fois | 8 | 9 | 10 |  |

* En utilisant seulement le tableau T\_occurences, construire un tableau TT contenant les mêmes éléments que T mais ordonnés dans l’ordre décroissant :

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TT : | 10 | 8 | 8 | 6 | 6 | 6 | 6 | 2 | 2 | 2 | 1 |

### 5.4 Exercice 4

**L’addition**

|  |  |  |  |
| --- | --- | --- | --- |
| La somme de |  | est |  |

* Ecrire un algorithme qui permet de**:**
  1. Saisir deux matrices A et B de même taille L x C. L <= 100 et C <=100. Les deux matrices sont remplis par des nombres aléatoires entre 1 et 1000 (*utiliser la fonction Random (1,1000)* )
  2. Calculer dans la matrice R (de taille L x C) le résultat de l’opération a.A+b.B. Avec a et b deux entier à saisir par l’utilisateur.

### 5.5 Exercice 5

Soit une matrice M de n lignes et m colonnes à valeurs entières. Ecrire un algorithme qui permet de saisir la matrice sachant que n et m sont des valeurs positives. Il détermine et affiche l’indice de la ligne dont la somme de ses éléments est maximale.