– Algorithmique – TP N°4 – TABLEAUX et CHAINES DE CARACTERES –

I – Les tableaux à une dimension

I.1– Déclaration : En langage C, un tableau est déclaré selon la syntaxe suivante :

TypeDeDonnee nomTableau[taille] ;

Exemple : ***int tab[10] ;*** permet de déclarer un tableau de 10 entiers (***int***).

En général, la taille du tableau est définie par la directive ***#define,*** au début du programme, juste en dessous des directives ***#include***. Exemple : ***#define N 10*** (Pas de signe égal =, ni point-virgule **;**)

✓ La déclaration d'un tableau à une dimension réserve un espace de mémoire contigu (continu) dans lequel les éléments (cases) du tableau sont rangés les uns à la suite des autres.

✓ Chaque élément du tableau **tab** est accessible par : **tab[indice]** (Indice est une expression entière).

✓ Les indices commencent par **0** et vont jusqu'à **taille – 1.** C'est-à-dire indice = 0, 1, 2, …, taille – 1.

Remarque importante : Le langage C, ne vérifie pas si **0 ≤ indice ≤ taille – 1**. Si **indice < 0** ou **indice ≥ taille,** vous essayer d'accéder à une position invalide car en dehors du tableau et vous risquez : ① d'obtenir des résultats aléatoires, ② d'écraser d'autres données qui n'appartiennent pas au tableau, ou ③ votre programme est arrêté brutalement (sans avertissement) par le système d'exploitation (Windows).

I.2 – Initialisation : Un tableau peut être initialisé par une liste de valeurs entre deux accolades.

Exemple : Initialisation d'un tableau de 10 entiers. **int tab[10] = { 9, -5, 8, 17, 36, 47, -43, -3, 42, -1 } ;**

	0	1	2	3	4	5	6	7	8	9
tab	**9**	**-5**	**8**	**17**	**36**	**47**	**-43**	**-3**	**42**	**-1**

✓ On peut n'initialiser que les premières cases du tableau, les autres cases seront, automatiquement, initialisées à zéro. **Exemple** : Si on écrit : **int tab[10] = { 9, -5, 8, 17} ;** le tableau sera initialisé comme suit :

tab	**9**	**-5**	**8**	**17**	**0**	**0**	**0**	**0**	**0**	**0**

✓ Pour initialiser toutes les cases du tableau à zéro, on écrit alors : **int tab[10] = { 0 } ;**

II – Les matrices (Tableaux à deux dimensions)

Exemple : On peut déclarer une matrice de 3 lignes et 4 colonnes comme suit : **int mat[3][4] ;**

Les indices de lignes sont 0, 1 et 2. Les indices de colonnes sont 0, 1, 2 et 3. Cette matrice peut être initialisée comme suit : **int mat[3][4] = { {11, 12, 13, 14 }, {21, 22, 23, 24 }, {31, 32, 33, 34}} ;**

mat	0	1	2	3
0	11	12	13	14
1	21	22	23	24
2	31	32	33	34

Remarque : Les cases d'une matrice sont rangées en mémoire ligne par ligne. La matrice de l'exemple précédent est sauvegardée en mémoire comme un tableau simple :

mat	11	12	13	14	21	22	23	24	31	32	33	34

III – Les chaines de caractères

En langage C, une chaine est un tableau de caractères qui se termine par le caractère nul ('\0').

III.1 – Déclaration et initialisation

Une chaine ***str*** de 10 caractères est déclarée comme suit : **char str[10] ;**

La chaine **str** comportera au maximum 9 caractères. Le 10^{e} caractère est réservé pour le caractère nul.

Une constante du type chaîne de caractères est, en général, initialisée avec une suite de caractères entre deux apostrophes doubles. **Exemple** : **char str[10] = "Bonjour" ;** // ce qui donne le tableau suivant :

str	B	o	n	j	o	u	r	'\0'	'\0'	'\0'

✓ Il est possible d'initialiser une chaine sur plusieurs lignes (cas de textes longs).

Exemple : char str[100] = "Bonjour " "Tout " "Le "
"Monde. "
"Vous allez bien ?" ;

Cette chaine sera regroupée comme suit : "Bonjour Tout Le Monde. Vous allez bien ?".

III.2 – Lecture et écriture (gets, puts)

✓ Il est possible de lire une chaine de caractères ***str*** avec l'instruction scanf("%s", &str). Cependant, l'instruction scanf ne permet pas de lire les espaces. On utilise à la place, l'instruction **gets(str)**.

✓ Pour afficher une chaine ***str***, on peut utiliser l'instruction printf("%s", str), mais on préfèrera l'instruction **puts(str)** qui ajoute à l'affichage un saut de ligne après la chaine ***str***.

III.3 – La bibliothèque string.h

La bibliothèque **string.h** offre plusieurs fonctions pour manipuler les chaînes de caractères :

① **strlen**(str) : donne la longueur de la chaine ***str*** (le caractère '\0' n'est pas compté).

② **strcpy**(strDestination, strSource) : copie la chaine strSource dans la chaine strDestination.

③ **strcat**(strDestination, strSource) : permet de concaténer les deux chaines strDestination et strSource. C'est-à-dire, elle copie strSource à la fin de la chaine strDestination.

④ **strcmp**(str1, str2) : permet de comparer les deux chaines ***str1*** et ***str2***. Si ***str1*** est identique (est égale) à ***str2***, strcmp retourne la valeur zéro. Si ***str1*** est inférieure à ***str2***, strcmp retourne une valeur négative. Sinon, elle retourne une valeur positive.

⑤ **strchr**(str, car) : recherche la **première** occurrence du caractère ***car*** dans la chaine ***str***.

⑥ **strstr**(str1, str2) : recherche la première occurrence de la chaine ***str2*** dans la chaine ***str1***.

III.4 – La bibliothèque stdlib.h

La bibliothèque **stdlib.h** offre les fonctions **atoi**(str), **atol**(str), **atof**(str) qui permettent de convertir une chaine de caractères ***str*** vers un ***int***, ***long***, ***double***, respectivement, si la conversion est possible.

Exemple : **atoi**("100") retourne la valeur entière (***int***) 100.

IV – LES EXERCICES – LES TABLEAUX

Exercice 1 : Recherche du maximum

Soit ***TAB*** un tableau de ***N*** entiers (Prenons ***N = 10***, par exemple).

Q1) Ecrire un programme C permettant de saisir (lire) les éléments du tableau ***TAB*** puis de les afficher.

Q2) Modifier le programme précédent pour chercher le maximum dans le tableau et afficher sa position.

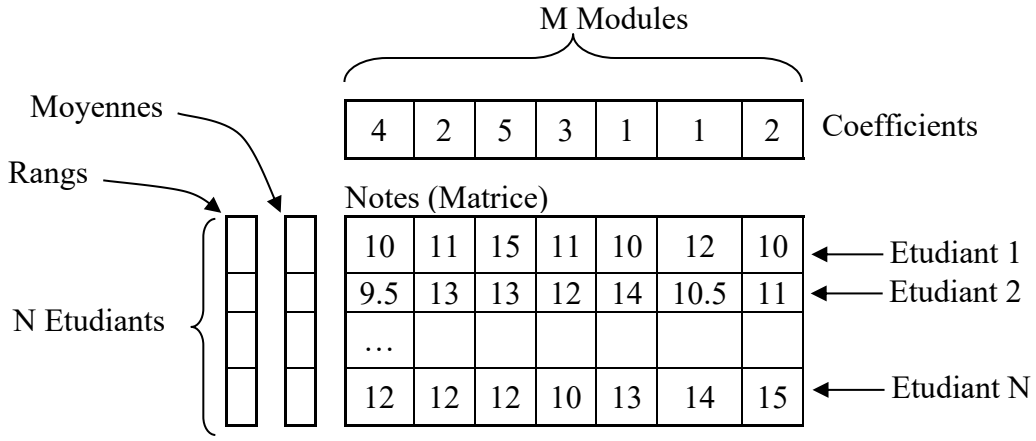
Exercice 2 : Moyenne du 1er semestre

L'objectif de cet exercice est de calculer la moyenne générale du 1er semestre de chaque étudiant d'une classe de 30 étudiants en 1ière Année Informatique. Ensuite, classer ces étudiants selon leurs moyennes.

Le tableau ci-dessous rappelle les modules suivis et leurs coefficients.

Modules	Analyse 1	Algèbre 1	Algorithmique 1	Structure Machine 1	Logiciels Libres	Anglais	Electricité
Coefficients	4	2	5	3	1	1	2

Pour modéliser ce problème, nous allons utiliser les 4 tableaux illustrés sur le schéma ci-dessous :



1) ***Coefficients :*** Un tableau d'entiers pour stocker les coefficients des modules,
2) ***Notes*** : Une matrice de nombres réels pour stocker les notes des étudiants,
3) ***Moyennes*** : Un tableau pour stocker les moyennes calculées de tous les étudiants,
4) ***Rangs*** : Un tableau d'entiers pour classer les étudiants selon leurs moyennes obtenues.

La déclaration des 4 tableaux peut se faire comme ceci :

```
#define     N     30     // défini au début du programme, juste en dessous des <include …>
#define     M     7

int     Coefficients[ M ] = {4, 2, 5, 3, 1, 1, 2}, Rangs[N] ;  // Déclarés dans int main ( )
float   Notes[ N ][ M ], Moyennes[ N ] ;
```

Travail demandé

Q1) Initialiser la matrice ***Notes*** (Lire les notes des étudiants). Pour valider les algorithmes écrits, on peut se limiter à 5 étudiants ***(Fixer N à 5)*** et 3 modules ***(Fixer M à 3)*** au lieu de 30 étudiants et 7 modules.

Q2) Chercher la note Max dans chaque module et afficher l'indice (N° ligne) de l'étudiant qui l'a obtenue.

Q3) Calculer la moyenne de chaque étudiant ***i*** et ranger la à la position ***i*** dans le tableau ***Moyennes***. Une fois toutes les moyennes calculées, afficher le tableau ***Moyennes***.

Q4) Calculer le nombre d'étudiants qui ont eu leur semestre. C'est-à-dire le nombre d'étudiants qui ont une moyenne supérieure ou égale à 10.

Q5) Chercher le major (celui qui a la plus grande moyenne). Afficher sa moyenne et son indice.

Q6) Classer les étudiants selon leurs moyennes générales avec la méthode suivante (**Tri par comptage**) :

- Pour chaque étudiant ***i*** compter le nombre d'étudiants qui ont des moyennes inférieures à la moyenne ***i*** et ranger le nombre trouvé dans le tableau ***Rangs*** à la position ***i***.
- Pour afficher les moyennes triés, il faut parcourir le tableau ***Rangs*** et afficher toutes les moyennes (ainsi que les indices de lignes) qui ont la valeur stockée dans le tableau ***Rangs*** égale à ***0, 1, 2*** ainsi de suite, jusqu'à ***N-1 (L'ordre est croissant)***.

Q7) Afficher tous les tableaux ensemble.

Exercice 3 : Chaines de caractères palindromes (Optionnel)

Un palindrome est une séquence dont l'ordre des éléments reste le même qu'on la parcourt du premier au dernier ou du dernier au premier. Ainsi, "ICI", "ELLE", "NON", "RADAR" sont des mots palindromes.

Q1) Ecrire un programme C qui vérifie si un mot est palindrome ou pas.

Q2) Modifier le programme précédent pour qu'il vérifie si une phrase est palindrome. Dans ce cas il faut ignorer (ou supprimer) les espaces. Exemples de phrases palindromes : « Ésope reste ici et se repose », « Engage le jeu que je le gagne », « L'âme sûre ruse mal », « Élu par cette crapule », …

****** Bon courage ******