TD 4 – Tableaux

Objectifs :

- *Créer un projet avec des fichiers sources et des fichiers en-tête*
- *Utiliser correctement des fonctions et des procédures déjà écrites*
- *Définir et appeler correctement des fonctions et des procédures*
- *Utiliser correctement des tableaux*

Exercice 1

1. Soit le tableau de valeurs suivantes :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
11	12	11	10	15	12	13	16	17	18	20	13	15	12	17

- Quelle est la valeur de l'indice minimal ?
- Quelle est la valeur de l'indice maximal ?
- À quel indice se trouve la valeur 18 ?
- À quels indices se trouve la valeur 12 ?
- Quelle valeur se trouve à l'indice 12 ?
- À quel indice se trouve la plus petite valeur ?
- À quel indice se trouve la plus grande valeur ?

2. On considère le tableau Tab2D à 2 dimensions suivant :

Tab2D

A	N	H	K	L
F	A	B	J	O
D	F	C	A	E

1. Quelle est la valeur de Tab2D [2][3],
2. Quelle est la valeur de Tab2D [0][2],
3. Donner la ligne et la colonne de la plus grande valeur,

Exercice 2

1. Initialisation d'un tableau à la compilation.

Dans un main :

- Déclarer et définir un tableau de 4 réels (float) en affectant à ces 4 éléments les valeurs 10.2, 20.7, -15.1, 30.9,
- Définir un réel que vous nommerez total,
- Calculer dans total la somme des éléments du tableau tab (bien évidemment vous utilisez une boucle !),
- Afficher total.

2. Initialisation d'un tableau à l'exécution.

Dans un main :

- Déclarer un tableau de 5 entiers,
- Déclarer un entier que vous nommerez total,
- Initialiser tous les éléments du tableau avec des nombres compris entre -10 et 10 à l'aide d'une saisie au clavier (utilisez pour cela la fonction saisieEntier qui vous a été donné pour le TD3 dans le fichier MesBibliotheques.c),
- Calculer dans total la somme des éléments positifs du tableau tab,
- Afficher total.

Exercice 3

Écrire un programme C permettant de :

- Déclarer et initialiser **à la compilation** un tableau de caractères de 10 éléments (vous pouvez mettre les valeurs que vous voulez pour initialiser votre tableau),
- Compter le nombre de caractères ' a ' dans ce tableau,
- Afficher ce tableau ainsi que le nombre de 'a' qu'il contient

Pour la suite de ce TD, lorsque vous devrez initialiser un tableau d'entier de N cases, vous pourrez utiliser cette procédure qui initialise un tableau avec des valeurs aléatoires qui seront toutes comprises entre 0 et valMax. Recopiez la dans vos projets.

```
void initTableauRandom (int tab [], int taille, int valMax)
{
    int i ;
    srand (time (NULL)) ;
    for (i=0; i<taille; i++)
        tab[i] = rand () % (valMax+1)  ;
}
```

Exercice 4

Écrivez une *procédure* qui affiche un tableau d'entiers de taille N. Écrivez également le programme principal qui :

- Déclare un tableau d'entiers T de taille 20,
- Initialise le tableau T avec des valeurs aléatoires comprises entre 0 et 50 (utilisez la procédure initTableauRandom pour cela),
- Affiche le tableau T.

Exercice 5

Écrivez une *fonction* qui retourne la position d'une valeur dans un tableau d'entiers de taille N (si la valeur est présente plusieurs fois c'est la position de la première valeur trouvée qui est retournée). Si la valeur n'est pas présente dans le tableau, votre fonction doit retourner -1.

Écrivez également le programme principal qui :

- Déclare 2 tableaux d'entiers de taille 10,
- Initialise un des tableaux (T1) avec des valeurs aléatoires comprises entre 0 et 50 (utilisez la procédure initTableauRandom pour cela),
- Initialise l'autre tableau (T2) à la compilation avec des valeurs que vous aurez choisies (c'est ce tableau que vous utiliserez quand vous testerez votre programme avec des assert)
- Affiche le tableau T1
- Demande à l'utilisateur de saisir une valeur entière
- Affiche la position de cette valeur dans le tableau T1.

Vous devez tester votre fonction avec assert avec des valeurs pertinentes pour le tableau T2.

Exercice 6

Écrivez une *fonction* qui retourne le nombre d'occurrences d'une valeur dans un tableau d'entiers de taille N. Écrivez également le programme principal qui :

- Déclare deux tableaux d'entiers de taille 10,
- Initialise un des tableaux (T1) avec des valeurs aléatoires comprises entre 0 et 50 (utilisez la procédure initTableauRandom pour cela),
- Initialise l'autre tableau (T2) à la compilation avec des valeurs que vous aurez choisies (c'est ce tableau que vous utiliserez quand vous testerez votre programme avec des assert)
- Affiche le tableau T1
- Demande à l'utilisateur de saisir une valeur entière
- Affiche combien de fois cette valeur est présente dans le tableau T1.

Vous devez tester votre fonction avec assert avec des valeurs *pertinentes* pour le tableau T1.

Exercice 7

Écrire une *fonction* qui permet de savoir si les éléments d'un tableau d'entiers sont triés dans l'ordre croissant. *Dès que la fonction peut déterminer que le tableau n'est pas croissant votre fonction doit s'arrêter*. Écrivez également le programme principal qui :

- Déclare plusieurs tableaux d'entiers que vous initialisez avec les valeurs que vous voulez à la compilation (il faut déclarer au moins un tableau croissant et un tableau non croisant),
- Affiche les tableaux
- Affiche si les éléments des tableaux sont triés dans l'ordre croissant.

Vous devez tester votre fonction avec assert avec des valeurs *pertinentes*.

Exercice 8

Écrivez la *fonction* qui retourne *la valeur minimum* contenue dans un tableau de taille N. Écrivez également le programme principal qui :

- Déclare plusieurs tableaux d'entiers de taille 10,
- Initialise un des tableaux (T1) avec des valeurs aléatoires comprises entre 0 et 50 (utilisez la procédure initTableauRandom pour cela),
- Initialise les autres tableaux à la compilation avec des valeurs que vous aurez choisies (ce sont ces tableaux que vous utiliserez quand vous testerez votre programme avec

des assert)
- Affiche le tableau T1
- Affiche l'élément minimum du tableau T1.

Vous devez tester votre fonction avec assert avec des valeurs *pertinentes*.

Exercice 9

Écrivez la *fonction* qui retourne *l'emplacement de la valeur minimum* contenue dans un tableau de taille N. Écrivez également le programme principal qui :

- Déclare plusieurs tableaux d'entiers de taille 10,
- Initialise un des tableaux (T1) avec des valeurs aléatoires comprises entre 0 et 50 (utilisez la procédure initTableauRandom pour cela),
- Initialise les autres tableaux à la compilation avec des valeurs que vous aurez choisies (ce sont ces tableaux que vous utiliserez quand vous testerez votre programme avec des assert)
- Affiche le tableau T1
- Affiche à quel emplacement se trouve l'élément minimum du tableau T1.

Vous devez tester votre fonction avec assert avec des valeurs *pertinentes* pour le tableau T2.

Exercice 10

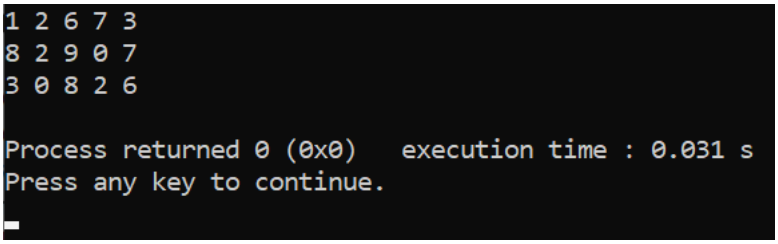
Écrivez la *fonction* qui calcule la *moyenne* des éléments d'un tableau. Écrivez également le programme principal qui :

- Initialise un tableau d'entiers de taille 10 (T1) avec des valeurs aléatoires comprises entre 0 et 50 (utilisez la procédure initTableauRandom pour cela),
- Affiche le tableau T1
- Affiche la moyenne des éléments du tableau T1.

Exercice 11

Écrivez dans un mail les instructions suivantes :

- Déclarer un tableau d'entiers de 2 dimensions : 3 lignes et 5 colonnes.
- Initialiser votre tableau à la compilation avec les valeurs que vous voulez.
- Afficher votre tableau de cette manière :



Exercice 12

Pour les exercices sur les tableaux à 2 dimensions, nous n'utiliserons pas de fonctions ou de procédures à cause de la manière dont CodeBlocks nous oblige à les passer en paramètres. Vous devrez donc le temps des exercices sur les tableaux à 2 dimensions les coder directement dans le main. Nous les utiliserons dans des fonctions/procédures un peu plus tard dans le module.

Écrivez le programme qui :

- Déclare un tableau d'entiers de 2 dimensions : 3 lignes et 5 colonnes.
- Initialise votre tableau à la compilation.
- Affiche votre tableau
- Affiche la valeur minimale contenue dans ce tableau

Exercice 13 :

Écrivez un programme qui détermine combien de fois une valeur donnée est présente dans un tableau de 2 dimensions. Écrivez les instructions qui :

- Déclare un tableau d'entiers de 2 dimensions : 3 lignes et 5 colonnes.
- Initialise votre tableau à la compilation.
- Affiche votre tableau
- Demande un nombre à l'utilisateur
- Affiche le nombre d'occurrences de ce nombre dans le tableau

Exercice 14

Écrivez la *fonction* plusGrandEcart qui détermine *la valeur du plus grand écart* entre 2 valeurs consécutives d'un tableau d'entiers de taille N. Écrivez également le programme principal qui :

- Déclare deux tableaux d'entiers de taille 10 chacun,
- Initialise un des tableaux (T1) avec des valeurs aléatoires comprises entre 0 et 50 (utilisez la procédure initTableauRandom pour cela),
- Initialise l'autre tableau (T2) à la compilation avec des valeurs que vous aurez choisies (c'est ce tableau que vous utiliserez quand vous testerez votre programme avec des assert)
- Affiche le tableau T1
- Affiche le plus grand écard entre 2 valeurs consécutives dans le tableau T1.

Vous devez tester votre fonction avec assert avec des valeurs *pertinentes* pour le tableau T2.

Exercice 15

Un palindrome est un mot qui peut se lire de la même manière qu'on le lise de gauche à droite ou de droite à gauche (par exemple ELLE, LAVAL sont des palindromes). Écrire une *fonction* qui détermine si un mot (c'est-à-dire un tableau de caractères) est un *palindrome*. Votre fonction doit retourner 1 si c'en est un et 0 sinon. Écrivez également le programme principal qui :

- Déclare et initialise à la compilation un tableau T de caractères (n'oubliez pas le caractère de fin de chaîne)
- Affiche le tableau T
- Détermine si le mot contenu dans le tableau T est un palindrome ou pas.

Vous devez tester votre fonction avec assert avec des valeurs *pertinentes*.

Exercice 16

Écrire la procédure qui *fusionne dans l'ordre croissant* deux tableaux T1 et T2 de dimension 1 triés dans *l'ordre* croissant et qui met le résultat dans un troisième tableau T3. Vos deux tableaux T1 et T2 ne sont pas forcément de la même taille.

Écrivez également le programme principal qui :

- Déclare deux tableaux d'entiers T1 et T2,
- Déclare un tableau T3 dont la taille est la somme des tailles de T1 et T2
- Initialise les 2 tableaux T1 et T2 à la compilation avec des valeurs que vous aurez

choisies

- Teste, avec assert, si les deux tableaux T1 et T2 sont bien triés dans l'ordre croissant
- Appelle la procédure qui fusionne T1 et T2 dans T3
- Affiche le tableau T3

Pour aller plus loin…

Année maximale de la population

On vous donne un tableau de nombres entiers en 2D, où chaque logs[i] = [birthi, deathi] indique les années de naissance et de décès de la ième personne.

La population d'une année x est le nombre de personnes vivantes au cours de cette année. La ième personne est comptée dans la population de l'année x si x est dans l'intervalle inclusif [birthi, deathi - 1]. Notez que la personne n'est pas comptée dans l'année de son décès. Retourner l'année la plus ancienne avec la population maximale.

Exemple 1 : Entrée : logs = [[1993,1999], [2000,2010]]

Sortie : 1993

Explication : La population maximale est de 1, et 1993 est l'année la plus ancienne avec cette population.

Exemple 2 : Entrée : logs = [[1950,1961], [1960,1971], [1970,1981]]

Sortie : 1960

Explication : La population maximale est de 2, et cela s'est produit dans les années 1960 et 1970. L'année la plus ancienne entre les deux est 1960.

Contraintes : 1 <= logs.length <= 100

1950 <= $naissance_i$ < $décès_i$ <= 2050