SÉRIE N° 12 : LES TABLEAUX/MATRICES (RÉVISION)

EXERCICE 1:

Ecrire un programme qui permet de décaler tous les éléments d'un tableau **T** de taille **N** : le second passe en premier, le troisième passe en second, etc.

Exemple :

Tableau initial de taille N=7 :

1	2	3	4	5	6	7

Tableau modifié (décalage à gauche) de taille 7

2	3	4	5	6	7	7

EXERCICE 2:

Ecrire un programme qui permet de

- calculer le ***nombre d'occurrences*** de 0 dans un tableau T de taille N
- supprimer les occurrences de 0.

EXERCICE 3:

Ecrire un programme qui élimine les doublons d'un tableau d'entiers de taille N. On considérera d'abord le tableau comme étant trié, puis on traitera le cas général.

EXERCICE 4:

Ecrire un programme qui permet de décaler tous les éléments d'un tableau **T** de taille **N** : le premier passe en second, le second passe en troisième, etc. Le dernier passe en premier.

Exemple :

Tableau initial de taille N=7 :

1	2	3	4	5	6	7

Tableau modifié (décalage à droite) de taille 7

7	1	2	3	4	5	6

EXERCICE 5:

Ecrire un programme qui saisit un entier x et un indice i et qui insert cet entier dans le tableau T de taille N à cet indice.

Exemple : pour i=1 et x=800

Tableau initial de taille N=7 :

1	2	3	4	5	6	7

Tableau modifié (décalage à droite) de taille 8

1	800	2	3	4	5	6	7

EXERCICE 6: recherche séquentielle

Ecrire un programme de ***recherche séquentielle*** d'un élément **x** entré par l'utilisateur dans un tableau des entiers de taille **N**.

- Le programme affiche la dernière position où se trouve x.
- Si **x** n'est pas dans le tableau, le programme affiche -1.

EXERCICE 7: recherche dichotomique

Ecrire un programme de ***recherche dichotomique*** d'un élément **x** entré par l'utilisateur dans un tableau de taille **N**. On supposera les éléments du tableau ordonnés par ordre croissant .

- Le programme affiche la première position où se trouve x.
- Si **x** n'est pas dans le tableau, le programme affiche -1.

Principe de la recherche dichotomique :

La recherche d'un élément dans un tableau non trié est linéaire (il faut parcourir en moyenne un nombre d'élément proportionnel au nombre d'élément du tableau et au pire les n éléments du tableau).
Si le tableau est trié (on suppose en ordre croissant) il est possible de faire une recherche beaucoup plus rapidement en utilisant une méthode dichotomique.
La recherche dichotomique dans un ensemble ordonné suppose de connaître le nombre d'éléments dans le tableau. Il suffit de regarder au milieu du tableau, de comparer ce que l'on trouve avec l'élément que l'on recherche. Si l'élément est le bon alors le recherche est finie sinon il faut recommencer avec la partie gauche ou la partie droite du tableau selon le résultat de la comparaison.

EXERCICE 8:

Ecrire un programme qui vérifie si un tableau **T** de taille **N** est symétrique ou non.

EXERCICE 9:

Ecrire les différentes parties d'un programme qui utilise deux tableaux, **T** et **P**, de **N** éléments de type **char** (où **N** est une constante qui sera fixée à 10 à l'aide le l'instruction **#define**) et permet de :

1. saisir au clavier les N éléments du premier tableau T.
2. afficher le premier tableau T (ses N éléments),
3. transférer les N éléments du premier tableau T dans le second P.
4. afficher le deuxième tableau P.
5. comparer les deux tableaux T et P et afficher un message indiquant que Les deux tableaux ne sont pas identiques, ou Les deux tableaux sont identiques!

EXERCICE 10:

Ecrire un programme qui calcule et affiche la somme de tous les éléments d'une matrice M de taille LxC.

EXERCICE 11:

Ecrire un programme qui permet de trier par ordre croissant une matrice M de taille LxC selon une colonne définie par l'utilisateur.

Exemple :

1	2	3	trié sur la colonne 2 =>	3	1	1
4	12	5		1	2	3
3	1	1		4	12	5

Prof : ZBAKH ABDEL ALI