TD n°3 : Tableaux

Partie I - Parcours de tableau

Exercice 1: Ecrire un algorithme qui
1) saisit un nombre de valeurs à lire,
2) crée un tableau d'entiers de la taille spécifiée,
3) lit une série d'entiers représentant des notes sur 20, et les rangee dans le tableau,
4) calcule la somme des éléments du tableau et la somme des indices,
5) recherche le plus grand élément du tableau et affiche son indice.

Exercice 2 (Nombre d'occurrences dans un tableau)
Ecrire une algorithme qui compte le nombre d'occurrences d'un caractère dans un tableau, c'est à dire le nombre de fois où un élément apparaît dans un tableau de caractères. Le caractère recherché et le tableau seront les deux paramètres de l'algorithme.

Exemples:

Caractère	tableau	résultat
'z'	{'a', 'b', 'c'}	0
'a'	{'a', 'b', 'c'}	1
'b'	{'a','b','c','b','d','b','a','d'}	3

Exercice 3 (Recherche dichotomique d'un élément dans un tableau trié)
La recherche dichotomique consiste à déterminer si un élément appartient à un tableau trié en divisant par 2 l'intervalle de recherche à chaque itération. Les opérations à effectuer lors d'une itération sont donc :
- déterminer la position de l'élément qui se trouve au milieu de l'intervalle courant ; si cet élément est l'élément recherché, l'algorithme se termine avec une réponse positive ;
- sinon
  - si l'élément du milieu est supérieur à l'élément recherché, poursuivre la recherche dans l'intervalle à gauche de l'élément du milieu ;
  - si l'élément du milieu est inférieur à l'élément recherché, poursuivre la recherche dans l'intervalle à droite de l'élément du milieu ;

On note g et d les indices délimitant l'intervalle de recherche à gauche et à droite respectivement.

1. Quelle condition permet de déterminer que l'élément recherché ne figure pas dans le tableau ?
2. Que valent :
   - l'indice de l'élément au milieu de l'intervalle ?
   - les bornes du nouvel intervalle de recherche si l'élément du milieu est supérieur à l'élément recherché ?
   - les bornes du nouvel intervalle de recherche si l'élément du milieu est inférieur à l'élément recherché ?
3. Écrivez un algorithme qui initialise un tableau de N nombres entiers avec des valeurs triées dans l'ordre croissant, puis qui demande à l'utilisateur de choisir une valeur à rechercher dans le tableau et affiche le résultat de la recherche.

Exercice 4 (Parcours partiel)
On souhaite remplacer dans un tableau `tabDest` les `nbVal` caractères enregistrés à partir de l'indice `indD` par les `nbVal` caractères d'un tableau `tabSource` enregistrés à partir de l'indice `indS` (voir exemple).

Ecrire un algorithme qui effectue ce traitement (parcours de deux sous-tableaux).

Exemple : nbVal : 3
`tabDest` : ABCDEFGHIJ, `indD` = 4
`tabSource` : MNOPQRST, `indS` = 5
Après l'appel de votre algorithme tabDest sera ABCQRSGHIJ, la sous-chaîne DEF ayant été remplacée par QRS.

Partie II - Inversion, test d'égalité et concaténation de tableaux

Exercice 5 (Inversion de tableau)
Soit tableau `Tab` d'entier de dimension `N`. Ecrire un algorithme qui inverse le tableau `Tab`

a. en passant par un tableau intermédiaire

b. sans utiliser de tableau d'aide (l'algorithme doit échanger les éléments du tableau à l'aide de deux indices qui parcourent le tableau en commençant respectivement au début et à la fin du tableau et qui se rencontrent en son milieu).

Exercice 6 (Test d'égalité entre deux tableaux)
On veut comparer deux tableaux `t1` et `t2` en testant l'égalité des valeurs contenues dans les deux tableaux.
Ainsi, `t1` et `t2` sont égaux si et seulement s'ils ont la même longueur et les éléments de même indice sont égaux, c'est à dire que `t1[i] == t2[i]` pour tout indice `i`.

Ecrire un algorithme qui réalise ce test d'égalité pour des tableaux de type `int[]`.

Exercice 7: Ecrire un algorithme qui prend deux tableaux de caractères et teste si tous les éléments du premier tableau apparaissent au moins une fois dans le deuxième tableau.

Exercice 8 (Concaténation de tableaux)
On appelle *concaténation* l'opération qui prend deux tableaux et calcule un tableau contenant les éléments du premier tableau, puis, à leur suite, les éléments du second tableau, dans le même ordre, mais avec un indice différent. Par exemple, `t3` ci-dessous est la concaténation de `t1` et `t2`.

`t1`

12	17	15	10

`t2`

13	14	11

`t3`

12	17	15	10	13	14	11

Exercice 9 (Crible d'eratostene)
Le but de cet algorithme est de trouver tous les nombres premiers jusqu'à un certain rang. Pour cela il procède en deux phases :

1) Se placer sur le prochain élément présent dans le tableau,

2) Enlever dans le reste de ce tableau tous les multiples de ce nombre.

3) Ecrivez cet algorithme.

Partie III - Tableaux à deux dimensions

Exercice 10 (Matrices)

1. Ecrire un algorithme permettant à un utilisateur de saisir une matrice carrée. Cette matrice aura une taille maximum donnée par une constante TAILLEMAX. L'utilisateur devra dans un premier temps dire la taille de la matrice qu'il veut saisir.
2. Compléter l'algorithme précédent afin d'afficher la matrice saisie par l'utilisateur.
3. Compléter l'algorithme précédent afin d'additionner deux matrices `M1` et `M2` dans une matrice `M3`. `M3(i,j)=`

`M1(i,j) + M2(i,j)`. On suppose `M1` et `M2` de même dimensions.

4. Compléter l'algorithme précédent afin de multiplier deux matrices `M1` et `M2` dans une matrice `M3`. `M3(i, j)= Somme [M1(k, j) * M2(i, k)], avec k = 1 à n`.

Exercice 11 (Carré Magique)
Ecrivez un algorithme qui vérifie si un carré est un carré magique (de taille quelconque).
Rappel : un carré est un carré magique si la somme de chacune des ligne = somme de chacune des colonnes = somme des diagonales.