TP 3 – Tableaux à une dimension

Exercice 1 – Inversion d'un tableau

Écrire une fonction ***void inverser (float t[ ], int n)*** qui permet d'inverser les éléments d'un tableau à une dimension.

Exemple : Si t = {5, 2, 7, 9, 1, 4}

Alors après inversion t = {4, 1, 9, 7, 2, 5}.

Exercice 2 – Inclusion entre deux tableaux

Écrire une fonction ***int inclusion (int *A, int n, int *B, int m)*** pour tester si les éléments d'un tableau B de taille m sont inclus dans le tableau A de taille n. La fonction retourne 1 si $B \subset A$ et 0 sinon.

Exemple : Si A = {5, 2, 7, 9, 1, 4, 2, 6} et B = {2, 4, 6} alors le résultat est 1.

Si A = {5, 2, 7, 9, 1, 4, 2, 6} et B = {2, 10, 6} alors le résultat est 0.

Exercice 3 – Problèmes des inversions dans un tableaux

Soit T un tableau à une dimension contenant une suite de n entiers distincts. On dit que les deux indices i et j forment une inversion dans T si i<j et T[i]>T[j]. Un problème intéressant consiste à déterminer le nombre d'inversions dans le tableau T. Le nombre d'inversions est le nombre de couples (i, j) pour lesquels i<j et T[i]>T[j].

1. Écrire une fonction qui retourne le nombre d'inversions dans un tableau T de taille n.
2. Générer dans la fonction principale main un tableau aléatoire de taille n (les éléments de 1 à n aléatoires), puis compter et afficher le nombre d'inversion pour n = 1000, 10000, 100000.
3. Observer l'évolution du temps d'exécution en fonction de n.

Exercice 4 – Schéma de Horner

On représente un polynôme réel $P(X) = a_0 + a_1X + a_2X^2 + \cdots + a_nX^n$ par un tableau à une dimension P de taille n+1 contenant ces coefficients $P = \{a_0 , a_1 , a_2 , \dots , a_n\}$

1. En utilisant la formule $P(X) = a_0 + a_1X + a_2X^2 + \cdots + a_nX^n$, écrire une fonction qui calcule la valeur de P(X) pour un polynôme P et une valeur X donnés en arguments.

En utilisant le schéma de Horner, on écrit le polynôme P sous la forme :

$$P(X) = a_0 + X(a_1 + X(a_2 + \cdots + X(a_{n-1} + a_nX) \dots))$$

2. En utilisant ce schéma, écrire une fonction qui calcule la valeur de P(X) pour un polynôme P et une valeur X donnés en arguments.

Estimez le nombre de multiplications effectuées par les deux fonctions et comparez.

Exercice 5 – Problème du drapeau hollandais

Le problème du drapeau hollandais est un problème de programmation, présenté par Dijkstra, qui consiste à réorganiser un ensemble d'éléments identifiés par leur couleur, sachant que seules trois couleurs sont présentes (par exemple, rouge, blanc, bleu, dans le cas du drapeau des Pays-Bas).

On suppose un tableau T de type **char** contenant les caractères **'R'** pour le rouge, **'W'** pour le blanc et **'B'** pour le bleu. On souhaite réarranger les éléments du tableau T comme suit : les éléments de couleur R au début, les éléments de couleur W au milieu, et les éléments de couleur B à la fin.

Exemple : Tableau initial : T = {W, R, R, W, B, W, B, B, R, W, R, W, B}

Tableau trié : T = {R, R, R, R, W, W, W, W, W, B, B, B, B}

Écrire une fonction pour réaliser ce tri.

Exercice 6 – Insertion dans un tableau trié

On considère un tableau t de taille n trié par ordre croissant.

Écrire une fonction qui permet d'insérer une valeur v dans t de telle sorte que t reste trié.

Exemple : Si t = {2, 4, 5, 7, 9, 11} et v = 6

Alors après insertion t = {2, 4, 5, 6, 7, 9, 11}.

N.B. *vous devez parcourir le tableau t une seule fois.*

Exercice 7 – Suppression dans un tableau

Écrire une fonction qui permet de supprimer toutes les occurrences d'une valeur v dans un tableau t de taille n.

Exemple : Si t = {2, 2, 1, 3, 5, 2, 2, 2, 9, 5, 3, 8, 2} et v = 2

Alors après suppression t = {1, 3, 5, 9, 5, 3, 8}.

N.B. *vous devez parcourir le tableau t une seule fois.*

Exercice 8 – Fusion de deux tableaux triés

1. On considère un tableau A de taille n trié par ordre croissant et un tableau B de taille m trié aussi par ordre croissant.

Écrire une fonction qui permet de fusionner A et B dans un tableau C de taille n+m de telle sorte que C soit aussi trié par ordre croissant.

Exemple : A = {2, 4, 5, 7, 9, 11} et B = {1, 3, 4, 6, 10, 14, 15}

Alors C = {1, 2, 3, 4, 4, 5, 6, 7, 9, 10, 11, 14, 15}

N.B. *vous devez parcourir les tableaux A et B une seule fois.*

2. Maintenant, on suppose que deux parties successives d'un tableau t sont triées par ordre croissant. La première partie délimitée par l'intervalle $[d, m[$ et la deuxième partie par l'intervalle $[m, f[$ avec $d, m, et\ f$ sont des indices donnés du tableau t.

Écrire une fonction ***void fusion (int *t, int d, int m, int f)*** qui permet de fusionner les deux parties du tableau de telle sorte la partie entre $[d, f[$ soit triée.

Exemple : t = {5, 4, **2, 6, 8, 10, 1, 3, 7, 11, 14,** 9, 12} et d = 2, m = 6, f = 11

Après fusion t = {5, 4, **1, 2, 3, 6, 7, 8, 10, 11, 14,** 9, 12}

3. Algorithme de trie par fusion :

L'algorithme de tri par fusion suit le principe « Divide and Conquer » (diviser pour mieux régner). Pour résoudre un problème à l'aide du principe Divide and Conquer, on exécute trois étapes d'une manière récursive : Divide – Conquer – Combine.

Dans le cas du tri par fusion, les 3 étapes sont :

- Divide : diviser le tableau de n éléments à trier en deux sous-tableaux de n/2 éléments.
- Conquer : trier les deux sous-tableaux récursivement à l'aide du tri par fusion.
- Combine : fusionner les deux sous-tableaux triées pour produire le tableau trié.

Implémenter l'algorithme de trie par fusion en langage C.