SOLUTION TD N°5 : LES VECTEURS ET LES MATRICES

Exercice 01 : Soit T un tableau à une dimension de N valeurs entières (avec N <= 100)

1. Ecrire un algorithme qui permet de lire ou de remplir le tableau (vecteur) T, et ensuite calcule la somme des valeurs de T.
2. Ecrire un algorithme qui permet d'afficher le contenu du tableau T.

Solution :

Algorithme :

```
Algorithme replir_tab;
Var T : Tableau[1..100] d'entiers ;
      n, i, S : entier ;
Début
   //1. Remplir le tableau
   Ecrire("Donnez la taille du tableau : ") ;
   Lire (n) ;
   Pour i de 1 à n Faire
    Lire(T[i]);
   Fpour

   //2. Calculer la somme
   S ← 0;
   Pour i de 1 à n Faire
    S ← S + i;
   Fpour
   Ecrire("La somme des éléments du tableau = ", S);

   //3. Afficher le tableau
   Pour i de 1 à n Faire
    Ecrire(T[i]);
   Fpour
Fin.
```

Exercice 02 : Soit "Etud" un tableau à une dimension qui contient les notes des étudiants dans une matière.

1. Ecrire un algorithme qui compte le nombre d'étudiants ayant une note:
   a. inférieur à 5
   b. supérieure ou égale à 5 mais inférieur à 10
   c. supérieure ou égale à 10

Solution :

Algorithme :

```
Algorithme notes;
Var Etud : Tableau[1..100] de réels ;
      n, i, S1, S2, S3 : entier ;
Début
   //1. Remplir le tableau
   .....

   //2. Calculer les sommes
   S1 ← 0; S2 ← 0; S3 ← 0;
   Pour i de 1 à n Faire
      Si  (T[i] < 5)  Alors
         S1 ← S1 + 1;
      Sinon
         Si  (T[i] <  10)  Alors
            S2 ← S2 + 1;
         Sinon
            S3 ← S3 + 1;
         Fsi
      Fsi
   Fpour
   Ecrire("La somme des éléments inférieur à 5 = ", S1);
   Ecrire("La somme des éléments supérieure ou égale à 5 et inférieur à 10 = ", S2);
   Ecrire("La somme des éléments supérieure ou égale à 10 = ", S3);
Fin.
```

Exercice 03 : Soit un tableau T de N valeurs entières (N <= 100)

1. Ecrire un algorithme qui permet de déterminer le maximum et le minimum du tableau T.

Solution :

Algorithme :

```
Algorithme tab_max_min;
Var T : Tableau[1..100] de entier ;
      n, i, min, max : entier ;
Début
   //1. Remplir le tableau
   .....

   //2. Trouver le min et le max
   min ← T[1];
   min ← T[1];
   Pour i de 2 à n Faire
      Si  (T[i] < min)  Alors
         min ← T[i];
      Fsi

      Si  (T[i] > max)  Alors
         max ← T[i];
      Fsi
   Fpour
   Ecrire("La min du tableau est ", min);
   Ecrire("La max du tableau est ", max);
Fin.
```

Exercice 04 : Soit un tableau T de N valeurs entières (N <= 100)

1. Ecrire un algorithme qui inverse le contenu du tableau T.
   a. En utilisant un tableau secondaire (temporaire)
   b. Sans utilisation d'un autre tableau (Travailler directement sur T)

Solution :

Algorithme « a » : en utilisant un 2ème tableau

```
Algorithme tab_inverse1;
Var T1, T2: Tableau[1..100] de entier ;
      i, j, n : entier ;
Début
   //1. Remplir le tableau T1
   .....

   //2. Inverser le tableau
   j ← n;
   Pour i de 1 à n Faire
      T2[j] ← T1[i];
      j ← j - 1;
   Fpour
Fin.
```

Algorithme « b » : sans utiliser un 2ème tableau

```
Algorithme tab_inverse2;
Var T : Tableau[1..100] de entier ;
      i, n : entier ;
Début
   //1. Remplir le tableau T1
   .....

   //2. Inverser le tableau

   Pour i de 1 à n DIV 2 Faire
      T[i] ← T[n-i+1];
   Fpour
Fin.
```

Exercice 05 :

Soit T un tableau à une dimension de N valeurs entières (N <= 100)

1. Ecrire un algorithme qui permet de rechercher une valeur entière **val** –donnée- dans le tableau **T** et retourne son indice si elle existe (retourne l'indice de la première occurrence) sinon, retourne une valeur négative.

Solution :

Algorithme :

```
Algorithme tab_search;
Var T : Tableau[1..100] de entier ;
      n, i, indice, val : entier ;
      trouve : booléen ;
Début

   //1. Lire la valeur à rechercher
   Lire(val) ;

   //2. Rechercher la valeur dans le tableau
   trouve ← faux;
   i ← 1;
   TQ (i <= n) ET (trouve = faux) Faire
      Si  (val = T[i])  Alors
         indice ← i;
         trouve ← vrai;
      Sinon
         i ← i + 1;
      Fsi
   Ftq

   Si  (trouve = vrai)  Alors   Ecrire (val,' existe à l''indice ', indice);
   Sinon    Ecrire (val,' n''existe pas');
Fin.
```

Exercice 06 : Soit deux vecteurs A[n] et B[n].

1. Ecrire un algorithme qui calcule le produit scalaire A*B. Le produit scalaire de deux vecteurs A et B est défini comme suit :

$$A * B = \sum_{i=1}^{n} (A[i] * B[i])$$

Exercice 07 : Soit un tableau T de N valeurs entières (N <= 100)

1. Ecrire un algorithme qui tri (ordonner les éléments) le tableau T selon un ordre (croissant ou décroissant).

Exercice 08 : Soit T un tableau d'entiers de taille égale à 100 contenants N valeurs (N<=100).

Ces valeurs sont triées selon l'ordre croissant.

1. Ecrire un algorithme qui insert une valeur entière dans le tableau T.

Solution :

Algorithme :

```
Algorithme insert_tab_ordonne;
Var T : Tableau[1..100] de entier ;
      n, i, indice, val : entier ;
Début
   //1. Lire la valeur à insérer
   Lire(val) ;

   //2. Décaler les cases à droite
   i ← n+1;
   TQ (i >= 2) ET (T[i-1] > val) Faire
      T[i] ← T[i-1];
      i ← i - 1;
   Ftq
   T[i] ← val;

   //3. Afficher le tableau après insertion
   Pour i de 1 à n Faire Ecrire(T[i]);

Fin.
```

Exercice 09 : Ecrire un algorithme qui fait la fusion de deux tableaux T1[n] et T2[m] triés selon l'ordre croissant, en un seul tableau T3 trié aussi selon l'ordre croissant.

Exemple :

5	7	11

T1

2	4	9	13

T2

2	4	5	7	9	11	13

T3

Solution :

Algorithme :

```
Algorithme tab_max_min;
Var T : Tableau[1..100] de entier ;
      n, i, min, max : entier ;
Début
    //1. Remplir le tableau
    .....


Fin.
```

Exercice 10 : Soit une matrice A [n, m] d'entières (N <= 100, M <=100)

1. Ecrire un algorithme qui permet de lire et de remplir la matrice A
2. Ecrire un algorithme qui permet d'afficher le contenu de la matrice A

Exercice 11: Soit T tableau -de valeurs réelles- à **n** lignes et **m** colonnes avec (n <= 50 et m <=50)

1. Ecrire un algorithme qui calcule la somme des éléments de ce tableau.

Exercice 12: Soit une matrice M [20, 20] de valeurs entières

1. Ecrire un algorithme qui recherche l'existence d'une valeur entière **val** donnée dans la matrice M et retourne sa position si elle existe.

Exercice 13: Soit une matrice A [n, n] une matrice carrée de valeurs entières avec n<=20

1. Ecrire un algorithme qui vérifie si la matrice A est triangulaire inférieure ou non ;

Exemple : A = $\begin{pmatrix} 5 & 24 & 10 & 2 \\ 0 & 39 & 8 & 13 \\ 0 & 0 & 33 & 7 \\ 0 & 0 & 0 & 11 \end{pmatrix}$ est triangulaire inférieure ;

Exercice 14: Soit une matrice A [n, n] une matrice carrée de valeurs entières avec n<=20

1. Ecrire un algorithme qui calcule et affiche la transposé da la matrice A

Exemple : A= $\begin{pmatrix} 3 & 4 & 10 \\ 5 & 0 & 1 \\ 12 & 13 & -8 \end{pmatrix}$ la transposé de A est A^t = $\begin{pmatrix} 3 & 5 & 12 \\ 4 & 0 & 13 \\ 10 & 1 & -8 \end{pmatrix}$

Exercice 15: Soit deux matrices T1 [n, m] et T2 [m, l] de valeurs réelles de dimensions inférieurs à 10x10.

1. Ecrire un algorithme qui calcule le produit de la matrice T1 fois la matrice T2 (T1 x T2)