

**Matière : INFORMATIQUE 2 : Algorithmique****ST : Série 4**

1° : Soit à remplir un vecteur avec les N premiers nombres entiers naturels, comme le montre l'exemple suivant :

	0	1	2	3	4	.....	.....	N-1
T=	2	4	6	8	10	....	....	....

***Corrigé***

- Apprendre à l'étudiant comment manipuler l'indice pour remplir un vecteur sans avoir à saisir les données :
- L'étudiant doit remarquer que le contenu de chaque élément est égal à :  
( La valeur de sa position + 1 ) \* 2, soit  
 $(i + 1) * 2$

Par exemple,  $T[N-1] = (N-1 + 1) * 2$ . Si  $N = 12$ ,  $T[N-1] = (12-1 + 1) * 2 = 12 * 2 = 24$ .

```

Algorithme Pair ;
Var
  N , i : Entier ;
  T : Tableau [ 50 ] de Réel ;
Début
  Répéter
    Ecrire ( ' Donnez la dimension <= 50 ' ) ;
    Lire ( N ) ;
  Jusqu'à N >= 1 et N <= 50 ;

  Pour i : = 0 à N - 1 faire
    Début
      T [ i ] : = ( i + 1 ) * 2 ;
      Ecrire ( ' T [ ' , i , ' ] = ' , T [ i ] ) ;
    Fin ;
Fin.

```

2° : Soit un vecteur V. Ecrire l'algorithme qui calcule :

- La somme des éléments de rang pair.
- Le produit des éléments de rang impair.
- Le nombre de valeurs nulles.

### Corrigé

- Apprendre à l'étudiant comment manipuler l'indice et les données d'un vecteur

```

Algorithme Pair ;
Var
  N , i , Nz : Entier ;
  S , P : Réel ;
  T : Tableau [ 50 ] de Réel ;
Début
  Répéter
    Ecrire ( ' Donnez la dimension <= 50 ' ) ;
    Lire ( N ) ;
  Jusqu'à N >= 1 et N <= 50 ;

  S := 0 ;
  P := 1 ;
  Nz := 0 ;
  Pour i := 0 à N - 1 faire
    Début
      // Au départ l'indice « i » est pair //
      // Saisie de la valeur de l'élément de rang pair //
      Ecrire ( ' Donnez la valeur de T [ ' , i , ' ] ' ) ;
      Lire ( T [ i ] ) ;
      Si T [ i ] := 0 alors
        Nz := Nz + 1 ; // Compte le nombre de valeurs = 0
        S := S + T [ i ] ; // Calcule la somme des éléments de rang pair

      // L'indice « i » devient impair //
      // Saisie de la valeur de l'élément de rang impair //
      i := i + 1 ;
      Si i <= N-1 alors // L'indice i doit être dans l'intervalle [0 , N-1]
        Début
          Ecrire ( ' Donnez la valeur de T [ ' , i , ' ] ' ) ;
          Lire ( T [ i ] ) ;
          Si T [ i ] := 0 alors
            Nz := Nz + 1 ; // Compte le nombre de valeurs = 0
            P := P * T [ i ] ; // Calcule le produit des éléments de rang impair
          Fin ;
        Fin ;
      Fin ;
    Ecrire ( ' La somme des éléments de rang pair = ' , S ) ;
    Ecrire ( ' Le produit des éléments de rang impair = ' , P ) ;
    Ecrire ( ' Le nombre de valeurs égales à zéro = ' , Nz ) ;
  Fin.

```

3° : Ecrire l'algorithme qui calcule le produit scalaire de deux vecteurs A et B.

$$Ps = \sum_{i=1}^{N-1} A[i] * B[i]$$

### Corrigé

- Apprendre à manipuler plusieurs vecteurs.

Algorithme P\_scalaire ;

Var

N , i : Entier ;

A, B : Tableau [ 50 ] de Réel ;

Ps : Réel ;

Début

Répéter

Ecrire ( ' Donnez la dimension <= 50 ' ) ;

Lire ( N ) ;

Jusqu'à N >= 1 et N <= 50 ;

// Remplir le vecteur A //

Pour i : = 0 à N - 1 faire

Début

Ecrire ( ' Donnez la valeur de A [ ' , i , ' ] ' ) ;

Lire ( A [ i ] ) ;

Fin ;

// Remplir le vecteur B et calculer en même temps le produit scalaire //

Ps : = 0

Pour i : = 0 à N - 1 faire

Début

Ecrire ( ' Donnez la valeur de B [ ' , i , ' ] ' ) ;

Lire ( B [ i ] ) ;

Ps : = Ps + A [ i ] \* B [ i ] // Calculer le produit scalaire : somme des produits des éléments de même rang

Fin ;

Ecrire ( ' Le produit scalaire de A et B = ' , Ps ) ;

Fin.

4° : Soit deux vecteurs T et W respectivement de dimension N et M ( $N \leq 25$ ,  $M \leq 16$ ). Ecrire l'algorithme qui fusionne T et W en un seul vecteur V tel que, aucun élément de V n'apparaît en double.

Comme le montre l'exemple suivant, la valeur « 7 », qui apparaît deux fois dans T ainsi que, les valeurs 15 et 3, qui sont communs à T et W, n'apparaissent qu'une seule fois dans le vecteur V.

T= 

3	7	15	4	20	17	7
---	---	----	---	----	----	---

W= 

15	6	3	18
----	---	---	----

V= 

3	7	15	4	20	17	6	18
---	---	----	---	----	----	---	----

### Corrigé

- On introduira dans cet exercice, la notion du booléen
- Familiariser l'étudiant à l'utilisation d'une boucle à l'intérieure d'une autre (boucles imbriquées) pour le préparer à la manipulation des matrices dans le prochain chapitre.

```

Algorithme Fusion_1 ;
Var
  N , M , i , j , k : Entier ;
  T : Tableau [ 25 ] de Réel ;
  W : Tableau [ 16 ] de Réel ;
  V : Tableau [ 41 ] de Réel ;
  Existe : booléen ;
Début
  Répéter
    Ecrire ( ' Donnez la dimension du vecteur T <= 25 ' ) ;
    Lire ( N ) ;
  Jusqu'à N >= 1 et N <= 25 ;

  Pour i : = 0 à N - 1 faire
    Début
      Ecrire ( ' Donnez la valeur de T [ ' , i , ' ] ' ) ;
      Lire ( T [ i ] ) ;
    Fin ;

  Répéter
    Ecrire ( ' Donnez la dimension du vecteur W <= 16 ' ) ;
    Lire ( M ) ;
  Jusqu'à M >= 1 et M <= 16 ;

  Pour i : = 0 à N - 1 faire
    Début
      Ecrire ( ' Donnez la valeur de W [ ' , i , ' ] ' ) ;
      Lire ( W [ i ] ) ;
    Fin ;

```

```

// Fusion à partir du vecteur T //
k := 0 ; // C'est l'indice du vecteur V
Pour i := 0 à N - 1 faire
    Début
        Existe := faux ;
        Pour j := 0 à k faire // Boucle pour parcourir le vecteur V
            Début
                Si T [ i ] = V [ j ] alors // Vérifie si l'élément de T existe déjà dans V //
                    Début
                        Existe := vrai ;
                        j := k ; // puisque l'élément existe dans V, on sort de la boucle
                    Fin ;
            Fin ;
        Si existe := faux alors // on vérifie si l'élément n'existe pas dans V
            Début
                V [ k ] = T [ i ] ;
                k := k + 1 ;
            Fin ;
        Fin ;

// Fusion à partir du vecteur W. Même boucle que la fusion à partir du T, sauf que //
// L'indice k du vecteur V ne doit pas être initialisé à zéro ! //
Pour i := 0 à M - 1 faire
    Début
        Existe := faux ;
        Pour j := 0 à k faire
            Début
                Si W [ i ] = V [ j ] alors
                    Début
                        Existe := vrai ;
                        j := k ;
                    Fin ;
            Fin ;
        Si existe := faux alors
            Début
                V [ k ] = W [ i ] ;
                k := k + 1 ;
            Fin ;
        Fin ;

// Affichage du vecteur V. La dimension de V et la somme des dimensions de T et W //
Pour i := 0 à ( N - 1 ) + ( M - 1 ) faire
    Ecrire ( ' V [ ' , i , ' ] = ' , V [ i ] ) ;
Fin.

```

Pour gagner en temps d'exécution et réduire la taille de l'algorithme, il est possible d'exécuter la fusion au moment de remplir un tableau.

```

Algorithme Fusion_2 ;
Var
  N , M , i , j , k : Entier ;
  T : Tableau [ 25 ] de Réel ;
  W : Tableau [ 16 ] de Réel ;
  V : Tableau [ 41 ] de Réel ;
  Existe : booléen ;
Début
  Répéter
    Ecrire ( ' Donnez la dimension du vecteur T ' ) ;
    Lire ( N ) ;
  Jusqu'à N >= 1 et N <= 25 ;
  k := 0 ;
  Pour i := 0 à N - 1 faire
    Début
      Ecrire ( ' Donnez la valeur de T [ ' , i , ' ] ' ) ;
      Lire ( T [ i ] ) ;
      Existe := faux ;
      Pour j := 0 à k faire // Boucle pour parcourir le vecteur V
        Début
          Si T [ i ] = V [ j ] alors // Vérifie si l'élément de T existe déjà dans V
            Début
              Existe := vrai ;
              j := k ; // puisque l'élément existe dans V, on sort de la boucle
            Fin ;
          Fin ;
        Si existe := faux alors // on vérifie si l'élément n'existe pas dans V
          Début
            V [ k ] = T [ i ] ;
            k := k + 1 ;
          Fin ;
        Fin ;
      Fin ;
    Répéter
      Ecrire ( ' Donnez la dimension du vecteur W ' ) ;
      Lire ( M ) ;
    Jusqu'à M >= 1 et M <= 16 ;
    Pour i := 0 à N - 1 faire
      Début
        Ecrire ( ' Donnez la valeur de W [ ' , i , ' ] ' ) ;
        Lire ( W [ i ] ) ;
        Existe := faux ;
        Pour j := 0 à k faire
          Début
            Si W [ i ] = V [ j ] alors
              Début
                Existe := vrai ;
                j := k ;
              Fin ;
            Fin ;
          Si existe := faux alors
            Début
              V [ k ] = W [ i ] ;
              k := k + 1 ;
            Fin ;
          Fin ;
        Fin ;
      Pour i := 0 à ( N - 1 ) + ( M - 1 ) faire
        Ecrire ( ' V [ ' , i , ' ] = ' , V [ i ] ) ;
      Fin.

```

5° : Soit un vecteur V de N valeurs. Ecrire l'algorithme qui calcule le nombre d'apparition du maximum dans le vecteur V.

### Corrigé

- Apprendre à l'étudiant la recherche du maximum dans un vecteur.

```

Algorithme Max ;
Var
    N , i , Nb : Entier ;
    T : Tableau [ 50 ] de Réel ;
    Max : Réel ;
Début
    Répéter
        Ecrire ( ' Donnez la dimension du vecteur T ' ) ;
        Lire ( N ) ;
    Jusqu'à N >= 1 et N <= 50 ;

    Pour i : = 0 à N - 1 faire
        Début
            Ecrire ( ' Donnez la valeur de T [ ' , i , ' ] ' ) ;
            Lire ( T [ i ] ) ;
        Fin ;

    // Initialisation //
    // Au départ, le Max reçoit la valeur du premier élément du vecteur T //
    Max := T [ 1 ] ;

    // Parcourir le vecteur à partir de la 2ème position pour chercher le nouveau Max //
    Pour i : = 1 à N - 1 faire
        Si T [ i ] > Max alors
            Max := T [ i ] ;

    // Calculer le nombre d'apparition du maximum dans le vecteur //
    Nb := 0 ;
    Pour i : = 0 à N - 1 faire
        Si T [ i ] = Max alors
            Nb := Nb + 1 ;
    Ecrire ( ' Le maximum apparaît dans le vecteur ' , Nb , ' fois. ' ) ;
Fin.

```

A noter, qu'il est possible de faire la recherche du max dans la boucle de saisie du vecteur.

```

Algorithme Max ;
Var
    N , i , Nb : Entier ;
    T : Tableau [ 50 ] de Réel ;
    Max : Réel ;
Début
    Répéter
        Ecrire ( ' Donnez la dimension du vecteur T ' ) ;
        Lire ( N ) ;
    Jusqu'à N >= 1 et N <= 50 ;

    Ecrire ( ' Donnez la valeur de T [ ' , 1 , ' ] ' ) ;
    Lire ( T [ 1 ] ) ;

    // Initialisation //
    // Au départ, le Max reçoit la valeur du premier élément du vecteur T //
    Max := T [ 1 ] ;

```

```

// Remplir le vecteur à partir de la 2ème position et chercher en même temps le Max //
Pour i : = 1 à N - 1 faire
    Début
        Ecrire ( ' Donnez la valeur de T [ ' , i , ' ] ' ) ;
        Lire ( T [ i ] ) ;
        Si T [ i ] > Max alors
            Max : = T [ i ] ;
        Fin ;

// Calculer le nombre d'apparition du maximum dans le vecteur //
Nb : = 0 ;
Pour i : = 0 à N - 1 faire
    Si T [ i ] = Max alors
        Nb : = Nb + 1 ;
Ecrire ( ' Le maximum apparaît dans le vecteur ' , Nb , ' fois. ' ) ;
Fin.

```

Il est aussi possible de calculer le nombre d'apparition du maximum dans la boucle de saisie du vecteur.

```

Algorithme Max ;
Var
    N , i , Nb : Entier ;
    T : Tableau [ 50 ] de Réel ;
    Max : Réel ;
Début
    Répéter
        Ecrire ( ' Donnez la dimension du vecteur T ' ) ;
        Lire ( N ) ;
    Jusqu'à N >= 1 et N <= 50 ;

    Ecrire ( ' Donnez la valeur de T [ ' , 1 , ' ] ' ) ;
    Lire ( T [ 1 ] ) ;

    // Initialisation //
    // Au départ, le Max reçoit la valeur du premier élément du vecteur T //
    Max : = T [ 1 ] ;
    Nb : = 1 ;

    // Remplir le vecteur à partir de la 2ème position et chercher en même temps le Max //
    Pour i : = 1 à N - 1 faire
        Début
            Ecrire ( ' Donnez la valeur de T [ ' , i , ' ] ' ) ;
            Lire ( T [ i ] ) ;
            Si T [ i ] >= Max alors
                Si Max : = T [ i ] alors
                    Nb : = Nb + 1
                Sinon
                    Début
                        Max : = T [ i ] ;
                        Nb : = 1 ;
                    Fin ;
            Fin ;
        Ecrire ( ' Le maximum apparaît dans le vecteur ' , Nb , ' fois. ' ) ;
    Fin.

```

6° : Ecrire l'algorithme qui permet de trouver le minimum et le maximum d'un vecteur et de permuter leur position, comme le montre l'exemple suivant :



T=	1	12	-3	0	5	39	32	19	69	14
T=	1	12	69	0	5	39	32	19	-3	14

**Corrigé***Apprendre la manipulation des indices*

Algorithme Min\_max ;

Var

N , i , ind\_max , ind\_min : Entier ;

T : Tableau [ 50 ] de Réel ;

Max , Min : Réel ;

Début

Répéter

Ecrire ( ' Donnez la dimension du vecteur T ' ) ;

Lire ( N ) ;

Jusqu'à N &gt;= 1 et N &lt;= 50 ;

Pour i : = 0 à N - 1 faire

Début

Ecrire ( ' Donnez la valeur de T [ ' , i , ' ] ' ) ;

Lire ( T [ i ] ) ;

Fin ;

// Initialisation //

// Au départ, le Max et le Min reçoivent la valeur du premier élément du vecteur T //

Max : = T [ 1 ] ;

Min : = T [ 1 ] ;

Ind\_max : = 1 ; // Représente l'indice où se trouve le Max

Ind\_min : = 1 ; // Représente l'indice où se trouve le Min

// Parcourir le vecteur à partir de la 2ème position pour chercher le Max et le Min, ainsi que leur position //

Pour i : = 1 à N - 1 faire

Si T [ i ] &gt; Max alors

Début

Max : = T [ i ] ;

Ind\_max : = i ;

Fin

Sinon

Si T [ i ] &lt; Min alors

Début

Min : = T [ i ] ;

Ind\_min : = i ;

Fin ;

// A la fin de la boucle on connaît le Max et le Min, ainsi que leur position : ind\_max et ind\_min //

// On permute leur position puis on affiche le vecteur.

T [ ind\_max ] : = Min ;

T [ ind\_min ] : = Max ;

Pour i : = 0 à N - 1 faire

Ecrire ( ' T [ ' , i , ' ] = ' , T [ i ] ) ;

Fin.

Email d'Ahmed-Nacer pour d'éventuels enrichissements, **CORRECTIONS** ou questions.[ahname06@yahoo.fr](mailto:ahname06@yahoo.fr)[ahname6@gmail.com](mailto:ahname6@gmail.com)