TD N°4 : Algorithmique et Programmation
LE – Informatique et Mathématique

Objectifs : Manipulation des Tableaux.

Exercice 1 :

Énoncé :

Écrire un algorithme qui déclare un **tableau** de **30 notes** dont on fait ensuite saisir les valeurs par l'utilisateur.

Solution :

La déclaration du tableau nommé **`Notes`** se fait via l'instruction **`tableau Notes[30] : réel`**, en spécifiant la taille du tableau entre crochets. On aura également besoin d'un entier (dans l'algorithme il s'agit de la variable **`i`**, qui va nous permettre de parcourir le tableau et d'avoir à chaque fois l'indice de chaque cellule du tableau.

```
Algorithme Notes
Variables i, Coeff, Compteur, Indice_Max, Indice_Min : entier
          tableau Notes[30] : réel
           Somme,MOY, Min, Max : réel

Début
      Pour i allant de 1 à 30 faire
            Écrire ("Saisie de la ", i, "notes")
            Lire (Notes[i])
      FinPour

Fin
```

Après avoir écrit l'algorithme, nous souhaitons ensuite effectuer l'ensemble des manipulations suivantes toujours dans le même algorithme :

1. ***Afficher la valeur de la 5ème note saisie par l'utilisateur.***

Pour afficher la cinquième note saisie, on doit la récupérer via son indice càd, **i=5,** comme suit :

```
Écrire ("Saisie de la cinquième note : ", Notes[5])
```

2. ***Afficher ensuite l'ensemble des notes saisies.***

Cette fois-ci, on doit parcourir l'ensemble du tableau afin de récupérer toutes les notes, on aura donc besoin d'une boucle.

```
Pour i allant de 1 à 30 faire
      Écrire ("T[", i, "]= ", T[i])
FinPour
```

3. ***Modifier la partie de la saisie des notes de sorte à ce qu'on teste la valeur entrée par l'utilisateur, celle-ci doit être comprise entre 0 et 20.***

Pour ce faire, il est clair que nous devons passer par une condition, mais pas une simple condition qui va nous permettre de tester la valeur de note, parce que de toute façon rien ne garantit que l'utilisateur va entrer la bonne valeur après un seul teste. Nous devons donc passer par une **boucle conditionnelle**, et tant que la valeur saisie pour la **i^{ème}** ne correspond pas à une valeur de note comprise entre 0 et 20, on ne doit pas passer à la saisie de la note suivante (càd. la **i+1ème** note).

```
Pour i allant de 1 à 30 faire
            Écrire ("Saisie de la ", i, "notes")
            Lire (Notes[i])
            TantQue (Notes[i]< 0 OU Notes[i]> 20)
                  Écrire ("Attention, Entrez une note comprise entre 0
                  et 20. Ressaisissez")
                  Lire (Notes[i])
            FinTantQue
FinPour
```

4. ***Calculer et afficher la somme des notes du tableau.***

Pour ce faire, on doit ajouter une autre variable pour y stocker la somme des notes. La déclaration se fait bien évidemment dans la partie **déclaration** plus-haut; **`Somme : réel.`**

```
Somme ← 0
Pour i allant de 1 à 30 faire
            Somme ← Somme + Notes[i]
FinPour

Écrire ("La somme des notes est : ", Somme)
```

5. ***Afficher les notes du tableaux après avoir multiplier toutes ces notes par un coefficient saisie par l'utilisateur.***

Un coefficient saisi par l'utilisateur au clavier, implique l'ajout d'une autre variable pour stocker ce coefficient. La déclaration se fait dans la partie **déclaration** plus-haut;

`Coeff : entier`

```
{Demander le coefficient}
Écrire ("Entrez le coefficient")
Lire (Coeff)
{Affichage des notes multipliées fois le coefficient}
Pour i allant de 1 à 30 faire
            Écrire ( Notes[i]*Coeff )
FinPour
```

6. ***Ajouter les instructions qui permettent de calculer et afficher la moyenne de ces notes.***

Nous avons déjà calculé la somme des notes (Question 4). Celle-ci est stockée dans la variable **Somme**. On passe donc directement à calculer la moyenne. Néanmoins, nous devons déclarer une autre variable pour y stocker la moyenne : **MOY : réel**

```
MOY ← Somme/30
Écrire ("La moyenne des notes est : ", MOY)
```

7. ***Compter et afficher ensuite le nombre des notes supérieures ou égales à 10.***

Pour compter le nombre des notes supérieures ou égales à 10, on doit parcourir toutes les notes du tableau pour tester pour chacune sa valeur si elle répond à la condition. Dans ce cas, cette note est comptée. Ceci est traduit par l'ajout d'une variable entière **Compteur** qui doit être incrémentée à chaque fois que la note répond à la condition.
Sans oublier la déclaration de la variable **Compteur** dans la partie dédiée aux déclarations.

```
Compteur ← 0
Pour i allant de 1 à 30 faire
      Si (Notes[i] >= 10) Alors
            Compteur ← Compteur + 1
      FinSi
FinPour
Écrire ("Nombre des notes supérieures ou égales à 10 est : ", Compteur)
```

8. ***Déterminer la note maximale, la note minimale ainsi que leurs positions dans le tableau.***

Pour ce faire, on déclare deux variables **Min** et **Max**, pour en stocker respectivement la valeur minimale et la valeur maximale du tableau. Dans un premier temps, les deux variables reçoivent la valeur de la première note du tableau ayant l'indice **i=1**. Ensuite, parcourir le tableau en comparant à chaque fois la valeur de la note courante **Notes[i]** avec la valeur de la variable **Max** et **Min.** La position de la valeur maximale (respectivement minimale) est stockée dans une autre variable **Indice_Max** (respectivement **Indice_Min**).

```
Min ← Notes [1]
Max ← Notes [1]
Pour i allant de 1 à 30 faire
{Pour retrouver la note maximale et son indice}
      Si (Notes[i] > Max) Alors
            Max ← Notes[i]
            Indice_Max ← i
      FinSi
{Pour retrouver la note minimale et son indice}
      Si (Notes[i] < Min) Alors
            Min ← Notes[i]
            Indice_Min ← i
      FinSi
FinPour
```

Exercice 2 :

Écrire un algorithme qui permet la saisie d'un tableau croissant (valeurs entières). Autrement dit, si la valeur courante à saisir est supérieure à la dernière valeur entrée au tableau, on l'enregistre sinon on redemande à l'utilisateur de refaire la saisie d'un nombre plus grand.

Corrigé :

Dans cet exercice, la taille du tableau n'est pas connue à priori, la déclaration donc du tableau se fait sans mentionner sa taille **`Tableau T [] : Réel`** .

Nous notons tout de même que tant qu'on n'a pas précisé le nombre d'éléments d'un tableau, d'une manière ou d'une autre, ce tableau est inutilisable. C'est à l'utilisateur donc de définir la taille du tableau.

Et après que l'utilisateur fournit la taille du tableau, on doit la fixer via une instruction de redimensionnement, il s'agit de la fonction Redim.

L'algorithme qui permet la saisie d'un tableau de valeurs ordonnées de manière croissante est le suivant:

```
Algorithme Tableau_Ordonné
Variables i, N : Entier
          Tableau T [] : Réel
Début
      Écrire ("Entrez la taille du tableau")
      Lire (N)
      Redim (T[N])
      Écrire ("Saisir le premier nombre du tableau")
      Lire (T[1])
      Pour i allant de 2 à N faire
            Répéter
                  Écrire ("Entrez le nombre numéro ", i)
                  Lire (T[i])
                  Si (T[i] < T[i-1]) Alors
                        Écrire (" Vous devez saisir un nombre plus grand")
                  FinSi
            Jusqu'à (T[i] >= T[i-1])
      FinPour
Fin
```

Remarque: Cette version accepte le fait d'avoir deux valeurs successives égales.

`(T[i]= T[i-1])`

Si vous souhaitez que votre algorithme ne prenne pas ce cas en considération, et qu'une seule valeur est permise (pas de doublons), il suffit d'exclure ce cas.

On reprend le même algorithme, en utilisant la boucle **TantQue** au lieu de **Répéter … Jusqu'à**

Remarque : De même, dans cette version, si vous souhaitez que votre algorithme n'accepte pas de doublons, il suffit de l'ajouter dans la condition du test pour la boucle.

```
TantQue (T[i] <= T[i-1])
```

```
Algorithme Tableau_Ordonné
Variables i, N, j : Entier
          Tableau T [] : Réel
          Val : Réel
Début
      Écrire ("Entrez la taille du tableau")
      Lire (N)
      Redim (T[N])
      Écrire ("Saisir le premier nombre du tableau")
      Lire (T[1])
      Pour i allant de 2 à N faire
            Écrire ("Entrez le nombre numéro ", i)
            Lire (T[i])
            TantQue (T[i] < T[i-1])
                  Écrire (" Vous devez saisir un nombre plus grand")
                  Lire (T[i])
            FinTantQue
      FinPour
Fin
```

Dans le même algorithme, ajouter les instructions qui permettent d'insérer une nouvelle valeur dans ce tableau trié tout en respectant l'ordre des valeurs du tableau.

L'insertion d'une nouvelle valeur dans le tableau revient à redimensionner celui-ci avant de procéder à l'insertion; càd. la taille du tableau va augmenter de un : **N+1**. Les instructions de redimensionnement sont les suivantes:

```
N ← N +1
Redim (T[N])
```

Nous passons ensuite à insérer la nouvelle valeur dans le bon endroit selon l'ordre du tableau, bien sûr après avoir récupéré la valeur au clavier.

À ce niveau, trois scénarios se posent, soit la valeur est plus petite que le premier élément du tableau l'ajout dans ce cas se fait au début du tableau càd à **T[1]**, ou bien comprise entre deux valeurs **T[i]** et **T[i+1]** du tableau, sinon le dernier cas, c'est que la valeur est supérieure à toutes les valeurs du tableau et donc à la valeur du **T[N-1]** du tableau, elle doit donc être insérée dans **T[N]**.

```
Écrire (" Donnez la valeur à insérer ")
Lire (Val)
```

```
Si (Val <= T [1]) Alors

     Pour i allant de N à 2 par pas -1 faire
     T [i] ← T [i-1]
     FinPour
     T [1] ← Val

Sinon
     Si (Val <= T [N-1]) Alors
       Pour i allant de 1 à N-1 faire
           Si (Val >= T [i] ET Val <= T [i+1]) Alors
                T [N] ← T [N-1]
                Pour j allant de N-1 à i+2 par pas -1 faire
                     T [j] ← T [j-1]
                FinPour
                T [i+1] ← Val
           FinSi
       FinPour
     Sinon
{Ce dernier cas correspond au cas où Val > T[N-1] }
     T [N] ← Val
     FinSi
FinSi
```

Et finalement on affiche le tableau ordonné résultat :

```
Écrire (" Le tableau ordonné résultat : ")

Pour i allant de 1 à N faire

   Écrire ("T[", i, "]= ", T[i])

FinPour
```

Exercice 3 :

Écrire un algorithme qui demande à l'utilisateur d'entrer 10 valeurs entières, ensuite, il les permute, de manière à placer le dernier élément à la place du premier et ainsi de suite.

Dans cet exercice, nous allons utiliser le principe de l'échange entre deux variables. Le principe est le suivant :

La valeur de **`T [1]`** doit être échangée avec celle de **`T [N]`**(soit **`N`** la taille du tableau, dans ce cas N=10), de même la valeur de **`T [2]`** doit être échangée avec celle de **`T [N-1]`** et ainsi de suite, jusqu'à ce qu'on parcourt le tableau à moitié N/2. Une valeur d'échange temporaire (**`tmp`**) est donc nécessaire pour sauvegarder la valeur à échanger.

L'algorithme est le suivant:

```
Algorithme Tableau_Permutation
Variables i, tmp, k : Entier
          Tableau T [10] : Réel

Début
     Pour i allant de 1 à 10 faire
          Écrire ("Entrez l'élément", i " du tableau")
          Lire (T [i])
     FinPour

     K ← 10

     Pour i allant de 1 à 5 faire
          tmp ← T [k]
          T [k] ← T [i]
          T [i] ← tmp
          k ← k-1
     FinPour

     Pour i allant de 1 à 10 faire
       Écrire ("T[", i, "]= ", T[i])
     FinPour

Fin
```

Exercice 4 :

Soit deux matrices A et B de taille (5,5).

1. Écrire l'algorithme permettant de saisir les valeurs des deux matrices A et B.
2. Ensuite dans le même algorithme, calculer et afficher la somme de ces deux matrices.
3. Calculer et afficher ensuite le produit des deux matrices.
4. Ajouter les instructions qui permettent de retrouver et afficher la plus grande valeur au sein de la matrice A ainsi que le position de cette valeur.

```
Algorithme Matrices
Variables i, j, r, iMax, jMax : Entier
          Tableau A[5][5] , B[5][5], som[5][5] , prod[5][5]: Entier
Début

{Saisie des valeurs de la matrice A }

      Pour i allant de 1 à 5 faire
            Pour j allant de 1 à 5 faire
                  Écrire ("Entrez le nombre n ", i, j, " de la matrice ")
                  Lire (A[i][j])
            FinPour
      FinPour

{Saisie des valeurs de la matrice B }

      Pour i allant de 1 à 5 faire
            Pour j allant de 1 à 5 faire
                  Écrire ("Entrez le nombre n ", i, j, " de la matrice ")
                  Lire (B[i][j])
            FinPour
      FinPour


{Calcule de la somme des deux matrices A et B dans la matrice som }

      Pour i allant de 1 à 5 faire
            Pour j allant de 1 à 5 faire
                  som[i][j] ← A[i][j]+ B[i][j]
            FinPour
      FinPour

{L'affichage de la somme des deux matrices som }

      Pour i allant de 1 à 5 faire
            Pour j allant de 1 à 5 faire
                  Écrire (som[i][j])
            FinPour
      FinPour
```

```
{Calcule du produit des deux matrices A et B dans la matrice prod }

      Pour i allant de 1 à 5 faire
            Pour j allant de 1 à 5 faire

                  prod[i][j] ← 0

                  Pour r allant de 1 à 5 faire

                     prod[i][j] ← prod[i][j]+ A[i][r] * B[r][j]

                  FinPour

            FinPour
      FinPour


{L'affichage du produit des deux matrices prod }

      Pour i allant de 1 à 5 faire
            Pour j allant de 1 à 5 faire
                  Écrire (prod[i][j])
            FinPour
      FinPour

{La recherche de la valeur maximale de la matrice A ainsi que sa position }

      iMax ← 1
      jMax ← 1

      Pour i allant de 1 à 5 faire
            Pour j allant de 1 à 5 faire
                  Si (A[i][j]>= A[iMax][jMax]) Alors
                        iMax ← i
                        jMax ← j
                  FinSi
            FinPour
      FinPour


{L'affichage de la valeur maximale de la matrice A ainsi que sa position }

   Écrire ("La plus grande valeur de la matrice A est : ", A[iMax][jMax])
   Écrire ("Il se trouve dans l'intersection de la ",iMax, "ème ligne et la
",jMax , "ème colonne")


Fin
```