Université Mohammed V de Rabat
Faculté des Sciences
Département d'Informatique

Correction de la série de TD 4

```
//Exemple de déclaration d'un tableau
Constantes  Max← 100 : Entier ;
Variable  Tableau T[Max] : Entier ;
//Exemple de déclaration d'une matrice
Constantes   MaxM←100 : Entier ; MaxN← 100 : Entier ;
Variable  Tableau  Matrice [MaxM][MaxN] : Entier ;
```

Exercice 1

```
Algorithme Remplir ;
Variables  i, n : Entier ;
Début
Ecrire ("Entrez la taille n" ) ; Lire(n) ;
Pour  i ← 0 à n-1 Faire
                Ecrire ("T[", i,"]=") ; Lire(T[i]) ;
Fin Pour
Fin
```

Exercice 2

```
Algorithme Afficher ;
Variables  i, n : Entier ;
Début
Ecrire ("Entrez la taille n" ) ; Lire(n) ;
        Pour i ← 0 à n-1 Faire
                Ecrire (T[i], " ") ;
        Fin Pour
Fin
```

Exercice 3

```
Algorithme MinimumMaximum ;
Variables i, vmin, vmax, n : Entier ;
Début
        //on suppose que le tableau est déjà rempli et est de taille n.
        vmin ← T[0] ; vmax←T[0] ;
        Pour  i←1 à n-1 Faire
                Si ( vmin > T[i] ) Alors vmin ←T[i] ;
                Fin Si
                Si ( vmax < T[i] ) Alors vmax←T[i] ;
                Fin Si

        Fin Pour
```

```
        Ecrire (" Le min du tableau est = ", vmin) ;
        Ecrire (" Le max du tableau est = ", v max) ;

Fin
```

Exercice 4

Principe de la recherche séquentielle :
Comparer x aux différents éléments du tableau jusqu'à trouver x ou atteindre la fin du tableau.

```
//Version avec Répéter… jusqu'à
Algorithme Recherche_seq ;
Variables i, n, x: Entier ;
        trouve : boolean ;
Début
i←-1; trouve←faux;
Repeter
        i←i+1;
TantQue (T[i] !=x) et (i < n-1) ;
Si (T[i] == x) Alors trouve ←Vrai ;
Fin Si
Si (trouve) Ecrire (" L'élément ", x, " se trouve dans le tableau T ") ;
Sinon  Ecrire (" L'élément ", x, " ne figure pas dans le tableau T ") ;
FinSi
Fin
```

```
//Version avec Tant que… Faire
Algorithme Recherche_seq ;
Varibles i, n, x : Entier
Début
i ←0 ;
Tant que (T[i] != x) ET (i <n-1) Faire
        i ← i+1 ;
Fin Tant que
Si (T[i]==x) Alors trouve ← Vrai ;
Sinon trouve ← Faux ;
Fin Si
Si (trouve) Ecrire (" L'élément ", x, " se trouve dans le tableau T ") ;
Sinon  Ecrire (" L'élément ", x, " ne figure pas dans le tableau T ") ;
Fin
```

Exercice 5

Principe de la recherche dichotomique :
Le but de la recherche dichotomique est de diviser l'intervalle de recherche par 2 à chaque itération. Pour cela, on procède de la façon suivante :
Soient premier et dernier les extrémités gauche et droite de l'intervalle dans lequel on cherche la valeur x, on calcule m, l'indice de l'élément médian :

```
        m←(premier + dernier) div 2 ;
```

Il y a trois cas possibles :
x < T[m] : l'élément x, s'il existe, se trouve dans l'intervalle [premier..m-1].
x > T[m] : l'élément x, s'il existe, se trouve dans l'intervalle [m+1...dernier].
x = T[m] : l'élément de valeur x est trouvé, la recherche est terminée.
La recherche dichotomique consiste à itérer ce processus jusqu'à ce que l'on trouve x ou que l'intervalle de recherche soit vide.

```
Programme Recherche_dich ;
Variables T : Tab ;
        n, x : Entier ;
premier, m, dernier : Entier ;
        trouve : Booleen
Début
premier ←0 ;
dernier ← n-1 ;
trouve←Faux ;
Répéter
m← (premier + dernier) div 2 ;

Si ( x< T[m] ) Alors dernier ←m – 1 ;
Sinon   Si ( x> T[m] ) Alors premier ←m + 1 ;
        Sinon trouve ←Vrai ;
        Fin Si
Fin Si

TantQue (non trouve) et (premier <= dernier) ;

Si (trouve) Ecrire (" L'élément ", x, " se trouve dans le tableau T ") ;
Sinon  Ecrire (" L'élément ", x, " ne figure pas dans le tableau T ") ;
FinSi

Fin
```

Exercice 6

```
Algorithme Fusion ;
Variables A, B, C : Tab ;
        i, j, n, m, k : Entier;
Début
k←0 ; i←0 ;
j←0 ;
Tant que (i < n) ET (j < m) Faire
  //On a l'élément  A[i] et plus petit que B[j] alors on ajoute A[i] dans C[k]
        Si(A[i] <=B[j]) Alors C[k]←A[i];
                        i ←i + 1;
                        k←k +1;
        Fin Si
```

```
//On a l'élément A[i] et plus grand que B[j] alors on ajoute B[j] dans C[k]
Si (B[j] <= A[i]) Alors C[k]←B[j] ;
                    j←j + 1 ;
                    k←k +1 ;
Fin Si
Fin Tant que
//Si on a encore des éléments dans A alors on les rajoute successivement dans C
Tant que (i < n) Faire
        C[k] ←A[i] ;
        i ←i + 1 ;
        k←k +1 ;
Fin Tant que
//Si on a encore des éléments dans B alors on les rajoute successivement dans C
Tant que (j < m) Faire
        C[k] ←B[j] ;
        j←j + 1 ;
        k←k +1 ;
Fin Tant que
Fin
```

Exercice 7

1. Remplissage d'une matrice

```
Algorithme Remplir ;
Variables  matrice : Mat ;
        i, j, n, m : Entier ;
Début
Pour i ←0 à n-1Faire
Pour j←0 à m-1Faire
        Ecrire ("Entrer un entier : ") ; Lire (T[i][j]) ;
Fin Pour
Fin Pour
Fin
```

2. Affichage d'une matrice

```
Algorithme Afficher ;
Variables matrice : Mat ;
        i, j, n, m : Entier ;
Début
Pour  i ←0 à n-1 Faire
Pour j←0 à m-1 Faire
        Ecrire (T[i][j]) ;
Fin Pour
Fin Pour
Fin
```

3. Somme de deux matrices

Principe:

Soient M1 et M2 deux matrices avec n ligne et m colonnes.

M3 = M1 + M2

```
Algorithme SomMat ;
Variables M1, M2, M3 : Mat ;
        i, j, n, m : Entier ;
Début
Pour i ←0 à n-1 Faire
Pour j ←0 à m-1 Faire
        M3[i][j] ←M1[i][j] + M2[i][j];
Fin Pour
Fin Pour
Fin
```

```
Algorithme  ProdMat ;
Variables M1, M2, M3 : Mat ;
        i, j, k, n, m : Entier;
Début
Pour  i← 0 à n-1 Faire
Pour  j←0 à m-1 Faire
        M3[i][j] ←0
        Pour k ←0 à m-1 Faire
                M3[i][j] ←M3[i][j] + M1[i][k] * M2[k][j] ;
        Fin Pour
Fin Pour
Fin Pour
Fin
```

Exercice 8

```
Algorithme occurrence ;
variables                   tableau   tab1[30] : entier ;
                    tableau  tabOccur[10] : entier ;
                   compteur, valMax, nombreMax : entier ;
Début
compteur←0 ;
TantQue (compteur < 30) faire
        Ecrire (" veuillez saisir un chiffre (0 à 9). ")
        Lire (tab1[compteur]) ;
        tabOccur[tab1[compteur]] ← tabOccur[tab1[compteur]]+1 ;
        compteur ← compteur+1 ;
FinTantQue
compteur ← 0 ;
```

```
valMax ← 0 ;
TantQue(compteur< 9) faire
Si (valMax < tabOccur[compteur]) alors  valMax← tabOccur[compteur] ;
                                        nombreMax ← compteur ;
FinSi
compteur← compteur + 1 ;
FinTantQue
Ecrire (" le nombre ", nombreMax,  " a été tapé", valMax, " fois ") ;
Fin
```