Université Félix Houphouët-Boigny
(UFHB)

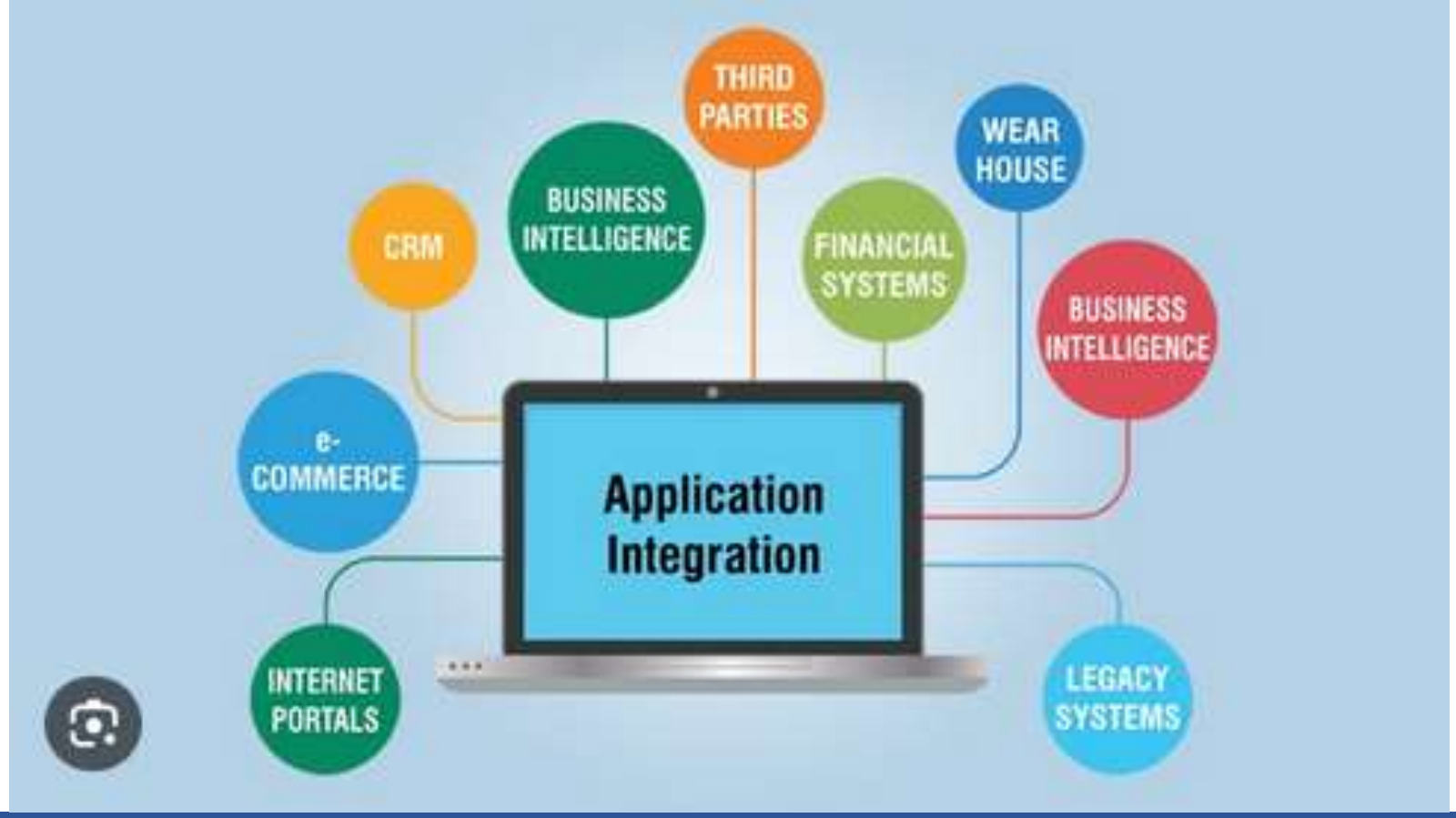
Cours d'algorithme avancée
L3-Info

Enseignant :

Dr GBAME Gbede Sylvain

Assistant, enseignant chercheur à l'UFHB

gbamegbedesylvain@gmail.com



MODULE : Algorithme avancée

PLAN

Chapitre 4
Les tableaux

Chapitre 4 : Les tableaux

4.0 Exemple Introductif

Écrire un algorithme qui permet de réaliser les tâches ci-dessous :

1. saisir 15 nombres ;
2. calculer leur somme
3. calculer leur moyenne ;
4. déterminer la parité de chacun de ces nombres ;
5. déterminer le plus grand de ces nombres ;
6. déterminer le plus petit de ces nombres.

Dans cet exercice, on doit manipuler individuellement ou collectivement 15 nombres par l'intermédiaire d'une variable pour chaque nombre. C'est "lourd". Il serait plus souple d'utiliser une variable différente des variables habituelles. En effet, cette nouvelle variable devra permettre de stocker plusieurs données, tandis que les variables habituelles ne peuvent stocker qu'une seule donnée.

Chapitre 4 : Les tableaux

4.0 Exemple Introductif

Les variables "**habituelles**" sont des variables simples (ou variables scalaires), tandis que les "***nouvelles*** **"** variables font partie des structures de données complexes.

La "***nouvelle***" variable est le **tableau**. Le tableau permet de stocker plusieurs données que l'on peut traiter de manière individuelle ou de manière collective.

Les applications des tableaux sont nombreuses. Par exemples :

- le stockage temporaire de données ;
- les calculs statistiques ;
- le calcul matriciel

Chapitre 4 : Les tableaux

4.1 Définition de base

On considère le tableau ci-dessous :

- Un ***tableau*** est un graphique constitué de **cellules** disposées en ***lignes*** et en ***colonnes*** ;
- Une ***ligne d'un tableau*** est constituée de deux segments horizontaux consécutifs ;
- Une ***colonne d'un tableau*** est constituée de deux segments verticaux consécutifs ;
- L'***intersection*** d'une colonne et d'une ligne est une ***case*** ou une ***cellule*** ;
- Le ***nombre de lignes*** et le ***nombre de colonnes*** constituent la ***taille*** de ce tableau ;
- La formule **nombre de lignes × nombre de colonnes** donne le **nombre de cellules** ;
- Le **nombre de cellules** d'un tableau caractérise aussi la **taille** de ce tableau.

Chapitre 4 : Les tableaux

4.1 Définition de base

Caractéristiques du tableau de l'exemple :

- 3 lignes ;
- 6 colonnes ;
- taille : 3 lignes et 6 colonnes notée tableau (3 × 6) ;
- d'où taille = 18 cellules.

Le tableau précédent admet une représentation qui ne tient compte que des cellules :

Un ***tableau de dimension 1*** (ou ***tableau à une dimension*** ou ***tableau unidimensionnel*** ou ***vecteur***) est un tableau constitué soit *d'une seule ligne et plusieurs colonnes*, soit de plusieurs lignes et une seule colonne. (Dans le premier cas, le tableau est horizontal ; dans le second cas, le tableau est vertical).

Chapitre 4 : Les tableaux

4.1 Définition de base

- Un ***tableau de dimension*** 2 (ou ***tableau à deux dimensions*** ou ***tableau bidimensionnel*** ou ***matrice***) est un tableau possédant des lignes et des colonnes. Le tableau de l'exemple est de dimension 2.

- Un ***tableau*** est une ***structure de données linéaire*** qui permet de stocker des données de ***même type***. Chaque donnée est repérée par un ou plusieurs ***indices*** qui indiquent la position de la donnée dans le tableau.

- Les données d'un tableau doivent être toutes du même type. C'est ce type commun qui est attribué au tableau. Donc le ***type d'un tableau*** est le type des données qu'il contient.

- Un ***indice*** est une variable de type entier qui sert à parcourir un tableau et à repérer une cellule du tableau.

Chapitre 4 : Les tableaux

4.1 Définition de base

- Dans un tableau, on stocke une donnée par cellule. Repérer une donnée revient aussi à repérer la cellule qui la contient;

Dans un tableau à une dimension, le repérage s'effectue à l'aide d'un seul indice, tandis que dans un tableau à deux dimensions, le repérage s'effectue à l'aide de deux indices;
Le tableau de l'exemple est un tableau à deux dimensions. Chaque cellule est située à l'intersection d'une ligne et d'une colonne.

Donc la cellule est repérée par deux indices :
un **indice ligne** et un **indice colonne**.

- Un tableau est une variable particulière. À sa déclaration, il possède :

✓ un identificateur ;
✓ un type ;
✓ un ou plusieurs indice(s).

- L'identificateur d'un tableau est le nom que l'on donne au tableau.

4.2 Les tableaux à une dimension

1. La déclaration

Avant de déclarer un tableau, il faut d'abord connaître le nombre de données à stocker et leur type commun. La déclaration permet de préciser l'*identificateur* du tableau, le *nombre maximal d'éléments* que le tableau va stocker, et le *type* du tableau, selon la syntaxe :

< **identificateur du tableau** > [**1 .. indice maximal**] : **tableau de** < **type de donnée**>

- < **identificateur du tableau** >

C'est le nom donné au tableau à sa déclaration.

- [**1 .. indice maximal**]

C'est l'intervalle de valeurs entières strictement positives que prendra la variable représentant l'indice.

4.2 Les tableaux à une dimension

La borne inférieure de cet intervalle vaut toujours 1 à la déclaration du tableau.
La valeur de la borne supérieure (*indice maximal*) dépend du nombre *maximal* de données à stocker dans le tableau. Ce qui signifie que la valeur de l'*indice maximal* indique le nombre maximal de données que l'on peut stocker dans un tableau de dimension 1.

Exemple: si le tableau doit contenir au maximum 15 valeurs, alors *indice maximal* vaut 15 ; d'où la notation **[1..15].**

- **tableau de < type de donnée >**

Cette notation indique le type commun des données que contiendra le tableau. Toutes les données d'un tableau doivent être du même type ou de *types compatibles*. Le tableau est alors du même type que les données (qu'il va stocker).

4.2 Les tableaux à une dimension

Exemples :

- si toutes les données sont de type entier, alors le tableau doit être de type entier ;
- si certaines des données sont de type entier et d'autres de type réel, alors le tableau doit être de type réel ;
- si certaines des données sont de type caractère et d'autres de type chaîne de caractères, alors le tableau doit être de type chaîne de caractères.

2. La saisie

Cette opération consiste à stocker une valeur dans chacune des cellules du tableau. On dit qu'on saisit le tableau ou qu'on renseigne le tableau. Cette opération se réalise de deux manières :

- soit on adresse directement chacune de ses cellules, dans un ordre quelconque ;
- soit on saisit le tableau de manière séquentielle (de la première cellule à la dernière).

4.2 Les tableaux à une dimension

On renseigne le tableau T en adressant directement chacune de ses cellules :

T [i] ← < **valeur** > // T[i] désigne la cellule d'indice i, et i varie de 1 à indice maximal

Exemple : On considère le tableau T [1..15] contenant des entiers

T [1] ← -3 // la cellule 1 reçoit la valeur -3 ;
T [8] ← 2015 // la cellule 8 reçoit la valeur 2015.

La saisie peut s'effectuer dans un ordre quelconque. Toutefois, il faut veiller à ne pas faire de "*débordement*", c'est-à-dire tenter de renseigner une ou plusieurs cellules qui ne font pas partie du tableau.

4.2 Les tableaux à une dimension

Exemple, les cellules T [16] et T [17] n'existent pas dans le tableau T [1..15].

- On ***renseigne le tableau T de manière séquentielle***, à l'aide de la boucle *Pour* car elle est mieux adaptée à cette opération :

```
Pour (i ← 1 à < indice maximal >) Faire
    < instructions >
    lire (T [i])
    < instructions >
FinPour
```

Exercice d'application 1 : Calcul de la moyenne d'un vecteur

Écrire un algorithme qui calcule la moyenne de 15 nombres saisis au clavier.

4.2 Les tableaux à une dimension

3. L'affichage

Cette opération est similaire à l'affichage du contenu d'une variable scalaire.
Elle consiste à afficher :

- soit le contenu d'une partie du tableau (le contenu d'une ou plusieurs cellules) ;
- soit le contenu de chaque cellule du tableau.

Considérons le tableau T [1: indice maximal] :

- On ***affiche le contenu de la cellule i de T***, à l'aide de l'instruction :

écrire (T [i])

4.2 Les tableaux à une dimension

- On ***affiche le contenu de chaque cellule de T***, à l'aide de la syntaxe générale :

```
Pour (i ← 1 à < indice maximal >) Faire
    < instructions >
    écrire (T [i])
    < instructions >
FinPour
```

Puisque l'affichage peut être sélectif, c'est-à-dire qu'il peut porter sur une partie des données du tableau, alors il faudrait adapter la condition régissant la structure *Pour* et les instructions contenues dans la structure *Pour*.

Avant d'afficher un tableau, il faut d'abord l'avoir saisi.

Il faudra bien sûr éviter les "*débordements*".

Remarque

T étant un vecteur, la notation T[i] désigne à la fois :

 la cellule d'indice i ;

4.2 Les tableaux à une dimension

Remarque

T étant un vecteur, la notation **T[i]** désigne à la fois :

- la cellule d'indice i ;
- et le contenu de cette cellule.

Exercice d'application 2 : Affichage d'un vecteur

a) Écrire un algorithme qui affiche les 15 nombres saisis au clavier.

b) Écrire un algorithme qui reçoit 15 nombres, puis n'affiche que ceux qui sont positifs ou nuls

4.2 Les tableaux à une dimension

4. Quelques opérations sur les vecteurs

On peut utiliser des vecteurs pour effectuer divers travaux tels que :

- ❖ les calculs statistiques
  - ➢ somme, produit, moyenne arithmétique
  - ➢ recherche d'extrémum
  - ➢ tris (tri croissant et tri décroissant)

- ❖ des calculs en géométrie analytique
  - ➢ produit scalaire à partir des coordonnées cartésiennes
  - ➢ distance entre deux points

- ❖ des opérations plus générales
  - ➢ recherche d'une valeur selon un critère quelconque
  - ➢ calculs divers

4.2 Les tableaux à une dimension

4.1. la recherche d'un élément dans un vecteur

On recherche un objet qui *pourrait* être dans l'un des nombreux casiers (ou tiroirs) d'un meuble. Ces casiers sont disposés les uns après les autres (en ligne ou en colonne).
On ouvre le premier casier, et on y recherche l'objet. S'il y est, on arrête la recherche ; sinon on continue jusqu'à atteindre le dernier casier.
Il y a deux résultats possibles : soit l'objet y est, soit il n'y est pas.

4.2 Les tableaux à une dimension

Exercice d'application 4 : Recherche d'une valeur dans un vecteur (cas 1)

Écrire un algorithme qui reçoit 15 nombres, puis qui détermine le nombre de fois qu'un nombre donné a été saisi.

❖ Analyse et méthode

D'abord, il faut saisir un vecteur de taille 15 de réels. Ensuite, on saisit le nombre à rechercher. Puis, on parcourt alors tout le vecteur, de la première cellule à la dernière. Chaque fois que l'on rencontre une occurrence du nombre recherché, on incrémente un compteur.
La structure *Pour* est la plus appropriée pour le parcours du vecteur.

4.2 Les tableaux à une dimensions

```
Algorithme Recherche_dans_vecteur_cas1
Var V[1:15] : tableau de réel
      nbrech : réel
      i, compteur : entier
Début
      écrire ("Recherche des occurrences d'un nombre dans une liste")
      écrire ("Saisie de 15 nombres")
      Pour (i ← 1 à 15) Faire
            écrire ("Donnez l'élément numéro ",  i,  ' : ')
            lire (V[i])
      FinPour
      écrire ("Donnez le nombre à rechercher : ")
      lire (nbrech)
      compteur ← 0
      Pour (i ← 1 à 15) Faire
            Si (V[i] = nbrech)
            Alors compteur ← compteur + 1
            FinSi
      FinPour
      Si (compteur = 0)
      Alors écrire (nbrech, "ne figure pas dans la liste de nombres saisis.")
      Sinon écrire ("Le nombre", nbrech, "a été saisi", compteur, "fois.")
      FinSi
Fin
```

4.2 Les tableaux à une dimensions

Exercice d'application 3 : Recherche d'une valeur dans un vecteur (cas 2)

Écrire un algorithme qui permet de vérifier si un nombre donné fait partie des 15 nombres qui ont été préalablement saisis.

❖ **Analyse et méthode**

On recherche un nombre particulier dans un vecteur de nombres. On parcourt alors le vecteur. On arrête la recherche dès que l'on trouve une occurrence du nombre recherché.
La structure *Tant Que* est appropriée pour cette recherche.

4.2 Les tableaux à une dimensions

```
Algorithme Recherche_dans_vecteur_cas2
Var V[1:15] : tableau de réel
      nbrech : réel
      i : entier
      trouve : booléen
Début
      écrire ("Recherche d'une occurrence d'un nombre dans une liste")
      écrire ("Saisie de 15 nombres")
      Pour (i ← 1 à 15) Faire
            écrire ("Donnez l'élément numéro ", i, ' : ')
            lire (V[i])
      FinPour
      écrire ("Donnez le nombre à rechercher : ")
      lire (nbrech)
      i ← 1
      trouve ← Faux
      TantQue ((i ≤ 15) ET (trouve = Faux)) Faire
            Si (V[i] = nbrech)
            Alors trouve ← Vrai
            Sinon i ← i + 1
            FinSi
      FinTantQque
      Si (trouve ≠ Faux)
      Alors écrire (nbrech, "ne figure pas dans la liste des nombres saisis.")
      Sinon écrire ("Le nombre", nbrech, "a été saisi en", i, "e position")
      FinSi
Fin
```

4.2 Les tableaux à une dimensions

4.2. la recherche des extrémums d'un vecteur

Il s'agit ici de rechercher la plus petite valeur et la plus grande valeur d'un vecteur.

Exercice d'application 6 : Recherche des extrémums d'un vecteur

Écrire un algorithme qui recherche le minimum et le maximum des 15 nombres saisis au clavier.

❖ **Analyse et méthode**

On parcourt le vecteur des nombres pour rechercher les extrémums. Le parcours s'effectue à l'aide de la boucle Pour. On y met en œuvre deux structures conditionnelles simples, l'une pour la recherche du minimum et l'autre pour la recherche du maximum.

4.2 Les tableaux à une dimensions

4.3. tri d'un vecteur

Un *tri* est une opération qui permet d'*organiser* les éléments d'un *ensemble fini* de données selon un *ordre* déterminé ou selon un ensemble de *critères*.
On se sert des vecteurs pour effectuer des opérations de tri sur des nombres, des caractères ou des chaînes de caractères. On réalise ces opérations à l'aide d'algorithmes spécifiques appelés *algorithmes de tri*.
Donc un *algorithme de tri* est un algorithme qui permet d'*ordonner* un *vecteur* d'éléments selon un *ordre* fixé.

Cette opération n'est pas traitée dans ce cours.

4.3 Les tableaux à deux dimensions

1. La déclaration

Cette opération permet de préciser l'identificateur, le nombre maximal de lignes, le nombre maximal de colonnes et le type du tableau, selon la syntaxe :

< nom du tableau > [1 .. indice ligne maximal, 1 .. indice colonne maximal] :
tableau de < type de donnée >

Exemple : On veut créer le tableau M de 3 lignes et 4 colonnes pour y stocker des réels :

- son nombre minimal de lignes est : 1 ;
- son nombre maximal de lignes est : 3, alors **indice ligne maximal = 3** ;
- son nombre minimal de colonnes est : 1 ;
- son nombre maximal de colonnes est : 4, alors **indice colonne maximal = 4**.

Puisque **M** est son identificateur, et qu'il contiendra des réels, alors on le déclare par l'instruction **M [1:3,1:4] : tableau de réel ou M [1..3,1..4] : tableau de réel**.

4.3 Les tableaux à deux dimensions

2. La saisie

Elle est similaire à celle des vecteurs, à la différence qu'ici toute cellule est repérée par deux indices : l'indice ligne et l'indice colonne.

Exemple :

Soient les variables de type entier i et j respectivement l'indice ligne et l'indice colonne du tableau **M [1:3,1:4]** de type réel : **i** varie de 1 à 3, tandis que j varie de 1 à 4.

L'opération de remplissage du tableau se réalise de deux manières :

- soit on adresse directement les cellules dans un ordre quelconque ;
- soit on saisit séquentiellement le tableau de la première cellule à la dernière.

❖ On saisit le contenu d'une cellule de M, à l'aide de l'instruction

M [i, j] ← < valeur >

4.3 Les tableaux à deux dimensions

On aura par exemples :

M [1,1] ← 25 // la cellule située à la ligne 1 colonne 1 reçoit le réel 25 ;

M [3,2] ← 2.5 // la cellule située à la ligne 3 colonne 2 reçoit le réel 2,5

La saisie peut s'effectuer dans un ordre quelconque. Mais il faut veiller à ne pas faire de "débordement", c'est-à-dire à donner des positions de cellules qui ne font pas partie du tableau.

Exemples : les cellules **M [4,2]** et **M [3,5]** n'existent pas dans le dans le tableau **M [1:3,1:4]**. Pourquoi ?

Si l'on désire saisir le tableau M de manière séquentielle de la première cellule à la dernière, on imbrique deux boucles Pour.

La boucle principale (boucle extérieure) parcourt le tableau en ligne tandis que la boucle intérieure parcourt le tableau en colonne.

4.3 Les tableaux à deux dimensions

Le parcours du tableau s'effectue selon le schéma ci-dessous, l'indice ligne i parcourant le tableau en ligne, tandis que l'indice colonne j parcourant le tableau en colonne :

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)

D'abord i "se positionne" sur la ligne 1 puis "s'y bloque", pendant que sur cette ligne, j parcourt les colonnes de la première à la dernière. C'est la raison pour laquelle la valeur de i sur la première ligne ne varie pas tandis que celle de j s'incrémente d'une colonne à une autre. Ensuite, dès que j atteint la dernière colonne, i passe à la deuxième ligne et "s'y bloque", en attendant que j parcourt toutes les colonnes à nouveau.

4.3 Les tableaux à deux dimensions

Ainsi de suite jusqu'à ce que i parcourt toutes les lignes.
Voici le bloc d'instructions pour la saisie séquentielle d'un tableau M à deux dimensions :

```
Pour (i ← 1 à < indice ligne maximal >) Faire
    Pour (j ← 1 à < indice colonne maximal >) Faire
        < instructions >
        lire (M [i, j])
        < instructions >
    FinPour j
FinPour i
```

4.3 Les tableaux à deux dimensions

3. L'affichage

Cette opération est similaire à l'affichage du contenu d'un tableau unidimensionnel.

On affiche :

- soit le contenu d'une cellule du tableau ;
- soit le contenu de tout ou partie du tableau.

❖ **Pour afficher le contenu de la cellule de la ligne i et de la colonne j** :

```
écrire (M [i, j])
```

4.3 Les tableaux à deux dimensions

❖ **Pour afficher le contenu de tout le tableau :**

```
Pour (i ← 1 à < indice ligne maximal >) Faire
    Pour (j ← 1 à < indice colonne maximal >) Faire
        < instructions >
        écrire (M [i, j])
        < instructions >
    FinPour j
FinPour i
```

Il faudra bien sûr éviter les "débordements".

4.3 Les tableaux à deux dimensions

Remarque

M étant un tableau à deux dimensions, la notation M [i, j] désigne à la fois :

- la cellule située à l'intersection de la ligne i et de la colonne j ;
- le contenu de cette cellule.

Exercice d'application 1 : Utilisation d'une matrice carrée

Écrire un algorithme qui remplit de réels une matrice carrée d'ordre 3, puis l'affiche.

Analyse et méthode

D'abord, une matrice est un tableau de dimension 2 ; elle est donc constituée de lignes et de colonnes. Ensuite, si le nombre de lignes est égal au nombre de colonnes alors cette matrice est carrée. Et enfin, l'ordre indique le nombre de lignes (ou de colonnes).

On doit donc d'abord déclarer un tableau de 3 lignes et 3 colonnes de type réel, ensuite le remplir et enfin l'afficher. Il est plus commode de renseigner le tableau de manière séquentielle. On imbrique alors deux structures Pour.

4.3 Les tableaux à deux dimensions

Remarque

M étant un tableau à deux dimensions, la notation M [i, j] désigne à la fois :

- la cellule située à l'intersection de la ligne i et de la colonne j ;
- le contenu de cette cellule.

Exercice d'application 1 : Utilisation d'une matrice carrée

Écrire un algorithme qui remplit de réels une matrice carrée d'ordre 3, puis l'affiche.

Analyse et méthode

D'abord, une matrice est un tableau de dimension 2 ; elle est donc constituée de lignes et de colonnes. Ensuite, si le nombre de lignes est égal au nombre de colonnes alors cette matrice est carrée. Et enfin, l'ordre indique le nombre de lignes (ou de colonnes).

On doit donc d'abord déclarer un tableau de 3 lignes et 3 colonnes de type réel, ensuite le remplir et enfin l'afficher. Il est plus commode de renseigner le tableau de manière séquentielle. On imbrique alors deux structures Pour.

4.3 Les tableaux à deux dimensions

La difficulté réside au niveau de la façon de guider l'utilisateur lors de la saisie.

En effet, il faut lui indiquer la position de la cellule active (c'est-à-dire celle qui va recevoir la donnée), dans une imbrication de boucle Pour, à l'aide de l'indice ligne et de l'indice colonne.

```
Algorithme SaisirMatrice_AfficherMatrice
Var M[1:3,1:3] : tableau de réel
    i, j : entier
Début
        écrire ("Saisie et affichage d'une matrice carrée d'ordre 3")
        écrire ("Saisie des éléments…")
        Pour (i ← 1 à 3) Faire
                Pour (j ← 1 à 3) Faire
                        écrire ("Donnez l'élément de la ligne ", i, " colonne ", j, " : ")
                        lire (M[i,j])
                FinPour j
        FinPour i
        écrire ("Saisie terminée !")
        écrire ("Vous avez saisi…")
        Pour (i ← 1 à 3) Faire
                Pour (j ← 1 à 3) Faire
                        écrire (M[i,j])
                FinPour j
        FinPour i
Fin
```

FIN DU COURS