Support de cours
Module : Algo 1
Chapitre 6: Les tableaux

Réalisé par:
Dr. Sakka Rouis Taoufik

1

Introduction

Supposons que nous avons besoin de déterminer à partir de 30 notes fournies en entrée, le nombre d'étudiants qui ont une note supérieure à la moyenne de la classe.

Pour faire face à ce problème, nous devons :

1. Lire les 30 notes
2. Déterminer la moyenne de la classe : m
3. Compter combien parmi les 30 notes sont supérieures à la moyenne m.

➔Il faut donc conserver les notes en mémoire afin qu'elles soient accessibles durant l'exécution du programme.

2

I. Introduction

Solution 1 : utiliser 30 variables réelles nommées x1, x2, …, x30

Cette façon de faire présente deux inconvénients :

- il faut trouver un nom de variable par note ;
- il n'existe aucun lien entre ces différentes valeurs. Or, dans certains cas, on est appelé à appliquer le même traitement à l'ensemble ou à une partie de ces valeurs.

Solution 2 : utiliser la notion de **tableau** qui consiste à :

- attribuer un seul nom à l'ensemble des 30 notes, par exemple Tnote,
- repérer chaque note par ce nom suivi entre crochets d'un numéro entre 1 et 30 : Tnote[1], Tnote[2], … Tnote[30].

3

II. Tableaux unidimensionnels

Un tableau à une dimension, appelé aussi **vecteur**, est une structure de données constituée d'un nombre fini d'éléments *de même type* et directement accessibles par leurs indices (ou indexes).

1) Déclaration

Syntaxe 1:

Nom_tab : **Tableau** [PremInd..DernInd] **de** Type_éléments

Exemple

Tnote : Tableau [1..5] de Réel

Schématiquement, ce tableau peut être représenté comme suit :

Nom : Tnote	100	200	300	400	500
Indice :	1	2	3	4	5
Contenu	T[1]	T[2]	T[3]	T[4]	T[5]

4

II. Tableaux unidimensionnels

Syntaxe 2:

Constantes

Nmax = 10

Types

Tab = Tableau de Nmax de réel

Variables

T : Tab

5

II. Tableaux unidimensionnels

2) Remplissage d'un tableau

Solution 1:

Un tableau peut être rempli *élément par élément* à l'aide d'une série d'affectations :

T[1] ←Valeur 1

T[2] ←Valeur 2

…

T[n] ←Valeur n

6

II. Tableaux unidimensionnels

Solution 2: Il est également possible de lire les éléments du tableau à partir du clavier grâce à une procédure :

Procédure remplir (Var T : tab; n: entier)
Variables
    i : Entier
Début
    Pour i **de** 1 **à** n **Faire**
        Écrire (”Entrer un entier : ”)
        Lire (T[i])
    FinPour
Fin Proc

7

II. Tableaux unidimensionnels

3) Affichage d’un tableau

Il est fortement recommandé d’afficher les éléments d’un tableau grâce à une procédure:

Syntaxe:

Procédure afficher (T : tab; n: Entier)
Variables
    i : Entier
Début
    Pour i **de** 1 **à** n **Faire**
        Écrire (T[i] , ”|”)
    FinPour
Fin Proc

8

III. Tableaux multidimensionnels

Les tableaux multidimensionnels sont des tableaux qui contiennent des tableaux.

Par exemple, le tableau bidimensionnel (appelé matrice de 3 lignes et 4 colonnes) suivant est en fait un tableau comportant 3 éléments, chacun d'entre eux étant un tableau de 4 éléments :

	1	2	3	4
1				
2				
3				

Chaque élément de la matrice est repéré par deux indices comme suite: M[i , j]

- le premier indique le numéro de la ligne
- le second indique le numéro de la colonne.

9

III. Tableaux multidimensionnels

1) Déclaration

Syntaxe1:

Variables

Matrice : Tableau[1..30, 1..40] de Réel

Syntaxe2:

Constantes

LMax = 30

CMax = 40

Types

Matrice : Tableau[1..LMax, 1..CMax] Réels

Variables

M : Matrice

10

III. Tableaux multidimensionnels

2) Remplissage

Il est fortement recommandé d'utiliser une procédure qui permet le remplissage d'un tableau multidimensionnel.

Syntaxe: cas d'un matrice simple

```
Procédure remplir (Var M : Matrice; n: Entier; m: Entier)
Variables
    i, j : Entier
Début
    Pour i de 1 à n Faire
        Pour j de 1 à m Faire
            Ecrire ("Entrer un entier :")
            Lire (M [i, j])
        FinPour
    FinPour
Fin
```

11

III. Tableaux multidimensionnels

3) Affichage

Il est fortement recommandé d'utiliser une procédure qui permet l'affichage d'un tableau multidimensionnel.

Syntaxe: cas d'un matrice simple

```
Procédure afficher (M : Matrice; n: Entier; m: Entier)
Variables
    i, j : Entier
Début
    Pour i de 1 à n Faire
        Pour j de 1 à m Faire
            Ecrire (M[i, j], " |")
        FinPour
        Ecrireln ()
    FinPour
Fin
```

12

IV. Exercices d'application

Exercice 1

- Déclarer le tableau T contenant au maximum 50 éléments de type entier.
- Proposer une procédure RemplirTab qui permet le remplissage du tableau T par n entiers compris entre 0 et 100.
- Proposer une procédure AfficheTab qui permet l'affichage des éléments du tableau.
- Proposer une fonction MinTab qui retourne le plus petit élément de ce tableau.
- Proposer un programme principale qui teste ces sous programmes.

13

IV. Exercices d'application

Exercice 2:

1) Ecrire la procédure suivante :

procédure Plateau (T : tableau[1..n] d'entiers; var P : tableau[1..n] d'entiers; var NbP : entier)

Cette procédure reçoit en paramètre un tableau T contenant n entiers, et remplit le tableau P avec **les indices de début des plateaux** de T.
La variable NbP contiendra le **nombre total de plateaux détectés**.

Rq. On appelle **plateau** une **séquence d'éléments consécutifs identiques** dans le tableau T.

Exemple
Pour T = [4, 4, 2, 2, 2, 5, 5] **On aura:** P = [1, 3, 6] et NbP = 3

2)Ecrire la fonction suivante:

fonction SommePlateau (T : tableau[1..n] d'entiers; pos: entier): entier

Cette fonction calcule et retourne la somme des éléments du plateau qui commence par la position pos.

14

IV. Exercices d'application

Exercice 3:

1) Ecrire la procédure suivante :

Procédure RemplirTab (varT: tableau[1..50] d'entiers ; n: entier)

Cette procédure permet de remplir le tableau T par n entiers.

2) Ecrire la procédure suivante :

Procédure afficherTab (T : tableau [1..50] d'entiers ; n : entier)

Cette procédure permet d'afficher les n premiers éléments du tableau T.

3) Ecrire la fonction suivante :

Fonction produitTab (T : tableau [1..50] d'entiers ; n : entier): entier

Cette fonction permet de calculer le produit des n premiers éléments du tableau T.

15

IV. Exercices d'application

Exercice 4:

Soit T un tableau contenant n éléments de type entier. On veut écrire une fonction dont l'entête sera :

Fonction recherche(T : Tab; n: Entier; x : Entier): Entier

Cette fonction retourne l'indice de la première occurrence de x dans T si x et appartient à T; sinon elle retourne la valeur 0.

Exercice 5: Soit T un tableau contenant n éléments de type entier et x un entier quelconque. Écrire une fonction qui retourne le nombre d'apparitions de x dans le tableau T. **Fréquence (T : Tab ; n: Entier ; x : Entier) : Entier**

16

IV. Exercices d'application

Exercice 6 : Éléments distincts

Écrire une fonction qui permet de remplir un tableau T par n entiers différents (chaque entier est présent une seule fois).

Exercice 7: Nombre d'éléments distincts

On se propose d'écrire un algorithme d'un programme qui saisit un entier n (10<n<=100) puis un tableau T composé de n entiers. Le programme calcule le nombre d'éléments distincts de T.

Exercice 8: Fréquence

Présenter d'une façon informelle et réaliser un algorithme permettant de compter la fréquence des éléments stockés dans un tableau. Ces éléments sont des entiers compris entre 1 et 100.

17