Leçon 4 : Les structures linéaires de tableau

Table des matières

Introduction

Comme en algorithmique, un tableau est une structure linéaire qui permet de conserver des valeurs de même type.

On distingue deux types de tableaux aussi en langage pascal à savoir :

- Les tableaux à une dimension ou vecteur
- Les tableaux à deux dimensions ou matrice

1- Tableau à une dimension

I

Définition : 1.1- Définition

Encore appelé vecteur, le tableau à une dimension est composé d'une seule ligne et plusieurs colonnes

Syntaxe : 1.2- Déclaration

1. **Cas 1 :**
   Var
   NomTab **: Tableau[Min..Max] de Type**
2. **Cas 2 :**
   Type
   t_tableau **= Tableau[Min..Max] de Type**
   Var
   NomTab **: t_tableau**

Remarque :

Min : premier élément du tableau

Max : dernier élément du tableau

Pour accéder à un élément du tableau il faut : NomTab [indice]. Indice : il s'agit du numéro de la colonne.

En Algorithme il existe deux manières de déclarer un tableau :

- Dans le cas 1, il s'agit de la déclaration classique (déclaration du tableau dans la var)
- et dans le second cas il faut déclarer le tableau comme un type et ensuite faire la déclaration de la variable avec le type créé.

Exemple :

var

Temp : Tableau[1..5] d'entier

ou

type

Tab = Tableau[1..5] d'entier

var

Temp : Tab ;

on vient de déclarer un tableau de nom Temp contenant 5 colonnes.

15	9	2	-9	-3

Temp

Temp[1] a pour valeur 15

Temp[2] a pour valeur 9

Temp[3] a pour valeur 2

Temp[4] a pour valeur -9

Temp[5] a pour valeur -3

Exemple : 1.3- Exemple

Énoncé :

Ecrire un algorithme permettant de renseigner 10 nombres pairs strictement supérieurs à zéro dans un tableau et de l'afficher.

Solution :

```
Algorithme nbrepair
type
tabpair = tableau[1..10] d'entier // déclare tout d'abord un type tableau
var
tnbrepair : tabpair  //déclaration de la variable avec le type créé précédemment
i : entier
Début
Pour i ← 1 à 10  faire //Permet de parcourir les dix colonnes du tableau
repeter
Afficher "Entrez un nombre pair strictement supérieur à zéro!"
Saisir tnbrepair[i]
jusqu'à ((tnbrepair[i] >0) and (tnbrepair[i] mod 2 = 0)) // vérifie que le nombre soit positif et pair
Finpour
//*** Affiche les informations du tableau ***
Pour i ← 1 à 10 faire
Afficher "tnbrepair[",i,"] =",tnbrepair[i])
Finpour
Fin
```

Application 1 :

II

Exercice

[solution n°1 p.14*]*

Donner la syntaxe de déclaration correcte :

- Typet_tableau : tableau[Min..Max] de type
- VarNomTab : t_tableau
- Typet_tableau : tableau[Min..Max] de typeVarNomTab : t_tableau

Exercice *[solution* n°2 p.14*]*

Énoncé :

Écrire un programme qui permet de remplir un tableau TAB de 20 éléments entiers quelconques non nul, Le programme devra ensuite parcourir le tableau pour rechercher le plus grand élément.

TAF : Veuillez remplir toutes les trous avec les valeurs qui conviennent.

NB : Toutes les réponses ne doivent pas avoir des espaces et elles doivent être en minuscule .

Solution :

tab

type

= de d'entier

var

: T_tab;

i, : entier

Début

Repeter

"Entrez votre nombre :"

jusqu'à

grd ←

Pour i ← a Faire

grd ←

Afficher "Le plus grand élément est :",

fin

2- Tableau à deux dimensions

Définition : 2.1- Définition

Encore appelé matrice, le tableau à deux dimensions est constitué de plusieurs lignes et plusieurs colonnes.

Syntaxe : 2.2- Syntaxe

Cas 1 :

Var

NomTab : Tableau[b1min..b1max,b2min..b2max] de type

Cas 2 :

type

t_tableau : Tableau[b1min..b1max,b2min..b2max] de type

Var

NomTab : t_tableau

Remarque :

b1min, b2min : premier élément du tableau

b1max, b2max : dernier élément du tableau

Pour accéder à un élément du tableau il faut : NomTab [i,j].

i : il s'agit du numéro de la ligne.

j : il s'agit du numéro de la colonne.

En langage algorithme il existe deux manières de déclarer un tableau :

- Dans le cas 1, il s'agit de la déclaration classique (déclaration du tableau dans la var)
- et dans le second cas il faut déclarer le tableau comme un type et ensuite faire la déclaration de la variable avec le type créé.

Exemple :

var

Mat : Tableau[1..5,1..3] d'entier

ou

type

Mat : Tableau[1..5,1..3] d'entier

var

Mat : Tab ;

on vient de déclarer un tableau de de 5 lignes et 3 colonnes.

2	5	6
9	-3	-6
0	5	14
3	11	-4
7	8	-2

Mat

Mat[1,1] a pour valeur 2

Mat[1,2] a pour valeur 5

Mat[1,3] a pour valeur 6

Mat[2,1] a pour valeur 9

Mat[2,2] a pour valeur -3

Mat[2,3] a pour valeur -6

Mat[3,2] a pour valeur 5

Mat[4,3] a pour valeur -4

Mat[5,1] a pour valeur 7

Exemple : 2.3- Exemple

Énoncé :

Ecrire un algorithme permettant de renseigner les 12 premiers nombres multiples de 3 dans une matrice de 3 lignes et 4 colonnes.

Solution :

Algorithme multitrois

var

matmulti : Tableau[1..3,1..4] d'entier

i,j,k : entier

Début

k ← 1 ;

Pour i ← 1 à 3 faire

Pour j ← 1 à 4 faire

```
matmulti[i,j] ← 3*k
k ← k + 1 ;
Finpour
Finpour
Pour i ←1 à 3 faire
Pour j ← 1 à 4 faire
Afficher "matmulti[",i,j,"] =",matmulti[i,j])
Finpour
Finpour
Fin
```

Application 2 :

IV

Exercice

[solution n°3 p.15*]*

Donner la syntaxe de déclaration correcte :

- VarTab : tableau[8..1,1..8] de réel
- typet_tab : tableau[1..8,1..8] de réelVarTab : t_tab
- typet_tab : tableau[1..8,1..8] de réelVarTab : t_tableau

Exercice *[solution* n°4 p.16*]*

Énoncé :

Écrire un programme permettant de saisir des données entières supérieur à zéro d'un tableau à deux dimensions (10,4), de faire leur somme, produit et moyenne(formater à 5 caractères à gauche et 2 caractères à droit) et de les afficher avec les résultats de calcul à l'écran.

TAF : Veuillez remplir toutes les trous avec les valeurs qui conviennent.

NB :

- Toutes les réponses ne doivent pas avoir des espaces et elles doivent être en minuscule.

Solution :

```
program matrice;
uses crt ;

mat : array[ ] of integer;
var
t :
i,j,som,prod
moy
begin
for i := 1 to do
begin
for j := 1 to do
begin
repeat
Entrez votre nombre :

until
end;
end;
som
prod
for i := 1 to do
begin
for j := 1 to do
begin
writeln(t[i,j]);
```

```
som :=som
prod :=prod
end;
end;
moy :=  10*4 ;
writeln('La somme est :',
writeln('Le produit est :',
writeln('La moyenne est :',
readkey;
end.
```

Solutions des exercices

> Solution n°1

Exercice p. 6

Donner la syntaxe de déclaration correcte :

- ○ Typet_tableau : tableau[Min..Max] de type
- ○ VarNomTab : t_tableau
- ◉ Typet_tableau : tableau[Min..Max] de typeVarNomTab : t_tableau

> **Solution** n°2

Exercice p. 7

Énoncé :

Écrire un programme qui permet de remplir un tableau TAB de 20 éléments entiers quelconques non nul, Le programme devra ensuite parcourir le tableau pour rechercher le plus grand élément.

TAF : Veuillez remplir toutes les trous avec les valeurs qui conviennent.

NB : Toutes les réponses ne doivent pas avoir des espaces et elles doivent être en minuscule .

Solution :

`Algorithme` tab

type

`T_tab` = `tableau[1..20]` de d'entier

var

`TAB` : T_tab;

i,`grd` : entier

Début

Repeter

`Afficher` "Entrez votre nombre :"

`Saisir` `tab[i]`

jusqu'à `tab[i]<>0`

grd ← `tab[1]`

Pour i ← `2` a `20` Faire

`si(grd<tab[i])alors`

grd ← `tab[i]`

`finsi`

`finsi`

Afficher "Le plus grand élément est :",`grd`

fin

> **Solution** n°3

Exercice p. 11

Donner la syntaxe de déclaration correcte :

- ○ VarTab : tableau[8..1,1..8] de réel
- ⊙ typet_tab : tableau[1..8,1..8] de réelVarTab : t_tab

- typet_tab : tableau[1..8,1..8] de réelVarTab : t_tableau

> **Solution** n°4

Exercice p. 12

Énoncé :

Écrire un programme permettant de saisir des données entières supérieur à zéro d'un tableau à deux dimensions (10,4), de faire leur somme, produit et moyenne(formater à 5 caractères à gauche et 2 caractères à droit) et de les afficher avec les résultats de calcul à l'écran.

TAF : Veuillez remplir toutes les trous avec les valeurs qui conviennent.

NB :

- Toutes les réponses ne doivent pas avoir des espaces et elles doivent être en minuscule.

Solution :

```
program matrice;
uses crt ;
type
mat : array[1..10,1..4] of integer;
var
t : mat;
i,j,som,prod:integer;
moy :real;
begin
for i := 1 to 10 do
begin
for j := 1 to 4 do
begin
repeat
writeln('Entrez votre nombre :');
readln(t[i,j]);
until t[i,j]>0;
end;
end;
som :=0;
prod :=1;
for i := 1 to 10 do
begin
for j := 1 to 4 do
```

```
begin
writeln(t[i,j]);
som :=som+t[i,j];
prod :=prod*t[i,j];
end;
end;
moy := som/10*4 ;
writeln('La somme est :',som);
writeln('Le produit est :',prod);
writeln('La moyenne est :',moy:5:2);
readkey;
end.
```