

Examen – Informatique 2

Exercice 01 : [5 pts]

Soit T, un tableau de N nombres entiers. Écrire un programme Pascal qui permet de diviser T en deux tableaux VP et VN : VP contient les valeurs positives et nulles et VN contient les valeurs négatives.

Exemple :



Exercice 02 : [8 pts]

Soit l'algorithme suivant :

Algorithme Exo2;

Variables

A : tableau [1..100,1..100] de réel;
 i, j, N : entier;
 S, M : réel;

Début

{*-*-* Les entrées *-*-*}
 Lire(N);

Pour i ← 1 à N faire

Pour j ← 1 à N faire

 Lire(A[i,j]);

Fin-Pour

Fin-Pour

{*-*-* Traitement *-*-*}

S ← 0;

Pour i ← 1 à N faire

 S ← S+A[i,N-i+1] ;

Fin-Pour

M ← S/N ;

{*-*-* Les sorties *-*-*}

Écrire(S:4:2, M:4:2);

Fin.

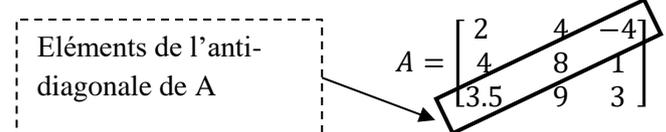
Questions :

- Traduire l'algorithme en Programme PASCAL.
- Dérouler le programme pour les valeurs suivantes : N = 3 et

$$A = \begin{bmatrix} 2 & 4 & -4 \\ 4 & 8 & 1 \\ 3.5 & 9 & 3 \end{bmatrix}$$

- Déduire ce que fait le programme ?
- Réécrire le programme en remplaçant la boucle for par la boucle while dans la partie de traitement.

Remarque :



Exercice 03 : [7 pts]

Soit le programme PASCAL suivant :

Program Exo3;

var

t, d : integer;
 v, vkm : real;
 Conversion (x: real): real ;

var

y : real;

Begin

 y := (x*3600)/1000 ;

End;

Begin

 read(d,t);
 v :=d/t ;
 vkm := conversion(v);
 writeln('La vitesse est : ', v:10:3 , ' m/s');
 writeln('La vitesse est : ', vkm:10:3, ' km/h');

End.

Questions :

- Identifier le type de sous-programme présent dans ce programme.
- Réécrire le sous-programme en complétant avec ce qui manque.
- Dérouler le programme pour d= 300 et t = 8.
- Déduire ce que fait le programme ?
- Réécrire le programme en transformant le sous-programme utilisé à l'autre type de sous-programme.

Bonne chance

Solution de l'examen – Informatique 2

Solution de l'exercice 01 : [5 pts]

Le programme Pascal qui permet de diviser T en deux tableaux VP et VN est le suivant: **(Solution 1)**

Programme PASCAL	
Program Classification ;	} 0.25
Var N,i,k,x : integer;	} 0.25
{Les variables k et x sont utilisées pour parcourir les éléments de VP et VN, respectivement. D'autres identificateurs (noms de variables) peuvent être utilisés}	
T,VP,VN : array [1..100] of integer ;	} 0.75 (0.25 pour chaque déclaration de tableau)
Begin	
{*-*-* Les entrées *-*-*	
Writeln('Donner la taille du vecteur T');	
Read(N) ;	} 0.25
Writeln('Donner les composantes du vecteur T');	
For i :=1 to N do	} 0.25
Read(T[i]) ;	
{*-*-* Traitement *-*-*	
x :=1 ; k :=1 ;	} 0.5 (0.25 pour l'initialisation de x et 0.25 pour celle de k)
For i :=1 to N do	} 0.25
if (T[i]>=0) then	} 0.5 (pour un if double ou pour deux if simples)
Begin	
VP[k] :=T[i] ;	} 0.5 (0.25 par instruction)
k :=k+1 ;	
End	
Else	
Begin	
VN[x] :=T[i] ;	} 0.5 (0.25 par instruction)
x :=x+1 ;	
End ;	
{*-*-* Les sorties *-*-*	
Writeln('Les composantes du vecteur VP sont : ');	
For i :=1 to (k-1) do	} 0.5 (0.25 pour write et 0.25 pour for qui varie jusqu'à k-1)
Write(VP[i]:4) ;	
Writeln;	
Writeln('Les composantes du vecteur VN sont : ');	
For i :=1 to (x-1) do	} 0.5 (0.25 pour write et 0.25 pour for qui varie jusqu'à x-1)
Write(VN[i]:4) ;	

End.

Ou bien (Solution 2)

Programme PASCAL

```
Program Classification ; } 0.25
Var
N,i,k,x : integer; } 0.25
{Les variables k et x sont utilisées pour parcourir les éléments de VP et VN respectivement. D'autres
identificateurs (noms de variables) peuvent être utilisés}
T,VP,VN : array [1..100] of integer ; } 0.75 (0.25 pour chaque déclaration de tableau)

Begin
  {*-**-* Les entrées *-**-*}
  Writeln('Donner la taille du vecteur T');
  Read(n) ; } 0.25
  Writeln('Donner les composantes du vecteur T');
  For i :=1 to N do } 0.25
    Read(T[i]) ;

  {*-**-* Traitement *-**-*}
  x :=0 ; k :=0 ; } 0.5 (0.25 pour l'initialisation de x et 0.25 pour celle de k)
  For i :=1 to N do } 0.25
    if (T[i]>=0) then } 0.5 (pour un if double ou pour deux if simples)

      Begin
        k :=k+1 ;
        VP[k] :=T[i] ; } 0.5 (0.25 par instruction)
      End
    Else
      Begin
        x :=x+1 ;
        VN[x] :=T[i] ; } 0.5 (0.25 par instruction)
      End ;

  {*-**-* Les sorties *-**-*}
  Writeln('Les composantes du vecteur VP sont : ');
  For i :=1 to k do } 0.5 (0.25 pour write et 0.25 pour for qui varie jusqu'à k)
    Write(VP[i]:4) ;
    Writeln;
  Writeln('Les composantes du vecteur VN sont : ');
  For i :=1 to x do } 0.5 (0.25 pour write et 0.25 pour for qui varie jusqu'à x)
    Write(VN[i]:4) ;

End.
```

N.B: Les messages et les commentaires ne sont pas notés.

Solution de l'exercice 02 : [8 pts]

1. Traduire l'algorithme en Programme PASCAL. (3.5 pts)

Algorithme	Programme PASCAL
Algorithme Exo2; variables A : tableau [1..100,1..100] de réel; i, j, N : entier; S, M : réel; Début {*-**-* Les entrée *-**-*} Lire(N); Pour i ← 1 à N faire Pour j ← 1 à N faire Lire(A[i,j]); Fin-Pour Fin-Pour {*-**-* Traitement *-**-*} S ← 0; Pour i ← 1 à N faire S ← S+A[i,N-i+1] ; Fin-Pour M ← S/N ; {*-**-* Les sorties *-**-*} Ecrire(S:4:2, M:4:2); Fin.	Program Exo2; } 0.25 var } 0.25 A:array [1..100,1..100] of real; } 0.25 i,j,N:integer; } 0.25 S, M : real; } 0.25 Begin {*-**-* Les entrée *-**-*} Read(N); } 0.25 For i:=1 to N do } For j:=1 to N do } 0.75 Read(A[i,j]); {*-**-* Traitement *-**-*} S:=0; } 0.25 For i:=1 to N do } 0.5 S:=S+A[i,N-i+1]; M:=S/N; } 0.25 {*-**-* Les sorties *-**-*} Write(S:4:2, M:4:2); } 0.25 End.

2. Dérouler le programme pour les valeurs suivantes : N = 3 et (2.5 pts)

$$A = \begin{bmatrix} 2 & 4 & -4 \\ 4 & 8 & 1 \\ 3.5 & 9 & 3 \end{bmatrix}$$

Instructions	Variables						Affichage																	
	N	i	j	A	S	M																		
Read(N);	3	/		/	/	/		0.25																
For i:=1 to N do For j:=1 to N do read (A[i,j]) ;	"	1 2 3	1 2 3	<table border="1" style="margin-left: 20px;"> <tr> <td></td> <td>j=1</td> <td>j=2</td> <td>j=3</td> </tr> <tr> <td>i=1</td> <td>2</td> <td>4</td> <td>-4</td> </tr> <tr> <td>i=2</td> <td>4</td> <td>8</td> <td>1</td> </tr> <tr> <td>i=3</td> <td>3.5</td> <td>9</td> <td>3</td> </tr> </table>		j=1	j=2	j=3	i=1	2	4	-4	i=2	4	8	1	i=3	3.5	9	3	/	/		0.5
	j=1	j=2	j=3																					
i=1	2	4	-4																					
i=2	4	8	1																					
i=3	3.5	9	3																					
S:=0;	"	/	/	"	0	/		0.25																
For i:=1 S:=S+A[i,N-i+1]; S:=0+A[1,3]; S:=0+(-4)=-4;	"	1		"	-4	/		0.25																
For i:=2 S:=S+A[i,N-i+1]; S:=-4+A[2,2]; S:=-4+8=4;	"	2	/	"	4	/		0.25																
For i:=3 S:=S+A[i,N-i+1]; S:=4+A[3,1]; S:=4+3.5=7.5;	"	3	/	"	7.5	/		0.25																
M:=S/N; M:=7.5/3=2.5;	"				"	2.5		0.25																
Write(S:4:2, M:4:2);	"						7.50 2.50	0.5																

3. Dédurre ce que fait le programme ? (0.75 pt)

Le programme calcule la somme et la moyenne des éléments de l'anti-diagonale de la matrice A. **0.75 (La formulation de cette phrase peut changer, cependant, les mots en surbrillance doivent apparaître)**

4. Réécrire le programme en remplaçant la boucle *for* par la boucle *while* dans la partie de traitement. (1.25 pt)

```
Programme PASCAL
Program Exo2;
var
  A:array [1..100,1..100] of real;
  i,j,N:integer;
  S, M : real;
Begin
  {*-**-* Les entrée *-**-* }
  Read(N);
  For i:=1 to N do
    For j:=1 to N do
      Read(A[i,j]);

  {*-**-* Traitement *-**-* }
  S:=0;
  i:=1;
  While (i<=N) do
  Begin
    S:=S+A[i,N-i+1];
    i:=i+1;
  End;
  M:=S/N;

  {*-**-* Les sorties *-**-* }
  Write(S:4:2, M:4:2);
End.
```

1.25 pt :
0.25 Initialisation.
0.25 Syntaxe de la boucle while
0.25 L'ajout du begin
0.25 Incrémentation
0.25 L'ajout du end

Solution de l'exercice 03 : [7 pts]

1. Identifier le type de sous-programme présent dans ce programme. (0.25 pt)

Le sous-programme est une **fonction 0.25**

2. Ré-écrire le sous-programme en complétant avec ce qui manque. (1 pt)

```
Programme PASCAL
Program Exo3;
var
  t, d : integer;
  v, vkm : real;
Function conversion (x: real): real ; }- 0.25 pour le mot-clé function
var
  y : real;
Begin
```

```

y := (x*3600)/1000 ;
conversion :=y; } 0.75
End;
Begin {Début du Programme Principal}
read(d,t);
v :=d/t ;
vkm := conversion(v);
writeln('La vitesse est : ', v:10:3, ' m/s');
writeln('La vitesse est : ', vkm:10:3, ' km/h');
End. {Fin du Programme Principal}

```

3. Dérouler le programme pour d= 300 et t = 8. (2.5 pts)

Instructions	Programme principal				La fonction			Affichage	
	t	d	v	vkm	x	y	conversion		
Read(d, t)	8	300	/	/					0.25
v :=d/t	"	"	37.5	/					0.5
vkm := conversion(v)	"	"	"	/					0.25
conversion(v) (l'appel à la fonction conversion avec le paramètre v=37.5) => La transmission des paramètres					37.5				0.5
y := (x*3600)/1000 ;						135			0.25
conversion :=y; → conversion = 135							135		0.5
vkm:= conversion(v)=135				135					0.25
writeln('La vitesse est : ', v:10:3, ' m/s');								La vitesse est : 37.5 m/s	
writeln('La vitesse est : ', vkm:10:3, ' km/h');								La vitesse est : 135 km/h	

4. Déduire ce que fait le programme ? (1 pt)

Le programme calcule la vitesse en m/s, puis fait appel à un sous-programme qui fait la conversion de cette vitesse en km/h. 1pt (La formulation de cette phrase peut changer, cependant, les mots en surbrillance doivent apparaître)

5. Réécrire le programme en transformant le type de sous-programme utilisé à l'autre type de sous-programme. (2.25 pts) (Solution 1)

```

Programme PASCAL

Program Exo3;
var
  t, d : integer;
  v, vkm : real;
Procedure conversion (x: real; var y:real) ;
Begin
  y := (x*3600)/1000 ;
End;
Begin {Début du Programme Principal}
read(d,t);
v :=d/t ;
conversion(v, vkm); } 0.75
writeln('La vitesse est : ', v:10:3, ' m/s');
writeln('La vitesse est : ', vkm:10:3, ' km/h');
End. {Fin du Programme Principal}

```

1.5 pt :
0.25 : Procedure
0.25 var, 0.25 y:real
0.25 Suppression du type
0.5 Suppression de conversion :=y

Ou bien (Solution 2)

Programme PASCAL

Program Exo3;

var

t, d : integer;

v, vkm, R : real; } **0.25**

Procedure conversion (x: real; var y:real); } **1.5 pt :**

Begin

y := (x*3600)/1000 ;

End;

Begin {Début du Programme Principal}

read(d,t);

v :=d/t ;

conversion(v, R); } **0.25**

vkm := R; } **0.25**

writeln('La vitesse est : ', v:10:3, ' m/s');

writeln('La vitesse est : ', vkm:10:3, ' km/h');

End. {Fin du Programme Principal}

0.25 : Procedure
0.25 var, 0.25 y:real
0.25 Suppression du type
0.5 Suppression de
conversion :=y