Examen – Informatique 2 (Corrigé Type)

Questions de cours : [3.5 points]

1- C'est quoi la différence entre les vecteurs et matrices ? Ils sont de quel type ? (**1 pts**)

Un vecteur est un tableau à **une dimension** vs Une matrice est un tableau à **deux dimensions** **0.25**

Un vecteur utilise **un seul indice** vs Une matrice utilise **deux indices** **0.25**

Un vecteur utilise **une seulle boucle** vs Une matrice utilise **une deux boucles** **0.25**

- Les vecteurs et les matrices sont de **type Tableau** ou de **type Indicé** (**type avec indice**) **0.25**

2- Donner la syntaxe générale pour déclarer une fonction. (**1 pts**)

function <nom_fonction> (<p1>:<type1>, <p2>:<type2>, …) : <type_fonction> ; **0.5**
var <v1>:<type1> ; <v2>:<type2> ; … ; <resultat>:<type_fonction> ; **0.25**
Begin
<instruction 1> ;
<instruction 2> ;
…
<nom_fonction> := <resultat> ; **0.25**
End ;

3- L'appel à un sous-programme doit vérifier quelquels conditions, quelles-sont ces conditions ? (**1.5 pts**)

Les condition qu'il faut vérifier lors de l'appel à sous programme :

0.5 - Le **nombre de paramètres effectifs doit être égale au nombres de paramètres formels**

0.5 - Le **type de chaque paramètre effectifs** doit être le même (ou compatible avec) le **type de paramètre formel correspondant**.

0.5 - **Respecter le type de passage : un paramètre formel doit être obligatoirement une variable lors du passage par variable, et il peut être une valeur fixe, une variable ou expression pour le passage par valeur.**

Exercice 01 : [9 points]

Partie A **:** Soit l'algorithme suivant :

```
Algorithme Exo01_A;
 Variables :      T : Tableau[1..100] de réel ;
    N, i, pos : integer ;    x:réel;
 Début
    Lire(N) ;
    Pour i ← 1 à N faire
       Lire( T[i] );
    Fin-Pour;
    Lire( X );     Lire (pos);

    Pour i ← Pos à N faire
      T[ N + Pos – i + 1] ← T[N +Pos–i] ;
    Fin-Pour;
    T[Pos] ← x ;      N ← N + 1;

     Pour i ← 1 à N faire
       Écrire( T[i] );
    Fin-Pour;
 Fin.
```

Questions :

1- Quelles sont les variables d'entrée et les variables de sorties de l'algorithme ? (**1 pts**)

2- Traduire l'algorithme en programme PASCAL. (**1 pts**)

3- Dérouler l'algorithme Pour : (**2.5 pts**)

N=4 , T=[15 , 2.5 , 6, 3] , X=-1.5 et Pos=2

4- Déduire ce que fait l'algorithme. (**0.5 pts**)

5- Récrire ***l'algorithme*** en remplaçant la boucle Pour par la boule Tant-que dans la partie Traitement. (**1 pts**)

1- Les variables d'entrées sont : **N , vecteur T (les cases de 1 à N), X et Pos** **0.5**

Les variables de sorties sont : **Vecteur T (les cases de 1 à N)** **0.5**

2- Traduction en programme PASCAL

```
Program Exo01_A;                                    } 0.25
 Var      T : Array [1..100] of real ;
    N, i, pos : integer ;    x:real;
 BEGIN
    read(N) ;                                       } 0.25
    for i:=1 to N do
       read( T[i] );

    Read( X );      Read(pos);

    for i:=Pos to N do                              } 0.25
      T[ N + Pos – i + 1] := T[N +Pos–i] ;

    T[Pos] := x ;      N := N + 1;

    for i:=1 to N do                                } 0.25
       write( T[i] );
 END.
```

3- Déourlement pour : N=4 , T=[15 , 2.5 , 6, 3] , X=-1.5 et Pos=2

Instructions	Variables					Affichage
	N	Pos	X	i	T	
Lire(N)	4	/	/	/	/	
Pour i ← 1 **à** N **faire** Lire(T[i]); **Fin-Pour;**	4	/	/	1 . .4	1 2 3 4 15 2.5 6 3	
Lire(x);	''		-1.5	''	''	
lire(pos)	''	2	''	''	''	
Pour i ← pos => i ← 2 T[N+Pos – i + 1] ← T[N+pos-i] T[4+2-2+1] ← T[4+2-2] T[5] ← T[4] = 3	''		''	2	1 2 3 4 5 15 2.5 6 3 3	
Pour i ← pos+1 => i ← 3 T[N+Pos – i + 1] ← T[N+pos-i] T[4+2-3+1] ← T[4+2-3] T[4] ← T[3] = 6	''	''	''	3	1 2 3 4 5 15 2.5 6 6 3	
Pour i ← pos+1 => i ← 4 T[N+Pos – i + 1] ← T[N+pos-i] T[4+2-4+1] ← T[4+2-3] T[3] ← T[2] = 2.5	''	''	''	4	1 2 3 4 5 15 2.5 2.5 6 3	
T[pos] ← x T[2] ← -1.5	''	''	''	''	1 2 3 4 5 15 -1.5 2.5 6 3	
N ← N+1 = 4+1 = 5	5	''	''	''	''	
Pour i ← 1 **à** N **faire** Écrire(T[i]); **Fin-Pour;**						15 -1.5 2.5 6 3

4- Déduire ce que fait l'algorithme

L'algorithme permet d'ajouter (d'insérer) la valeur de X dans le vecteur T à la position Pos.

5- Remplacer la boucle Pour par Tantque

```
Algorithme Exo01_A;
 Variables :      T : Tableau[1..100] de réel ;
    N, i, pos : integer ;    x:réel;
Début
    Lire(N) ;
    Pour i←1 à N faire
       Lire( T[i] );
    Fin-Pour;
    Lire( X );      Lire (pos);
    i ←Pos;
    Tant-que I <= N faire
      T[ N + Pos – i + 1] ← T[N +Pos–i] ;
      i ← i +1;
    Fin-Tant-que;
    T[Pos] ← x ;      N ← N + 1;

     Pour i←1 à N faire
       Écrire( T[i] );
    Fin-Pour;
Fin.
```

Partie B : Programme PASCAL

```
Program Exo01_B;
 Var      A : Array [1..100, 1..100] of integer;
          N, i, j : integer;
          S, nb : integer;
 BEGIN
    read(N) ;
    for i:=1 to N do
        for j:=1 to N do
            read( A[i, j] );

    S := 0;
    for i:=1 to N do
        if A[i, i] mod 2 = 0 then
            S := S + A[i, i];

    Nb := 0;
    for i:= 1 to N do
        if A[  i  ,  N-i+1   ] mod 2 <> 0 then
            Nb  :=  Nb + 1;

    write( S, nb );
END.
```

Ou bien

if …. **mod 2 = 1** then

Exercice 02 : [7.5 points]

1 – Programme PASCAL

```
Program Exo02;
 var n, m : real;
 function traitement(x:real) : real;
  Const PI = 3.14;
  Var  y : real;
 Begin
      y := x * PI / 180;
 traitement := y;
 End;
BEGIN
    Read( n );
    m := traitement(n);
    Write( m );
END.
```

2 – Le sous programme utilisé est une ***fonction***

3- Déroulement pour n=90

Instructions	Variables					Affichage
	P.P.		Traitement(fonction)			
	n	m	x	y	Traitement	
Read(n);	90	/				
m := traitement(n) Appel Traitement(n) => traitement(90) → Transmission de paramètre y := 90 * PI / 180 = PI/2 traitement := y;			90	PI/2	PI/2 = 1.57	
On remplace traitement(n) par PI/2 => m := PI/2		PI/2				
write(m);						PI/2 = 1.57

4 – Le programme permet de **convertir** une valeur d'un **angle** de **Dégré** vers **Radian**

5 – Réécrire le programme en remplaçant la fonction par procédure

```
Program Exo02_5;
 var n, m : real;
procedure traitement(x:real; var y : real);
  Const PI = 3.14;
                    Suppression de la variable local y
 Begin
      y := x * PI / 180;
 End;                Suppression de traitement := y;

BEGIN
    Read( n );
    traitement(n, m);
    Write( m );
END.
```