Solution : série 2

Exercice 1. ***** veuillez expliquer aux étudiants comment on a calculé les valeurs 7, 25 et 3.5

- L'algorithme affiche :

```
La somme de a et b = 7
La somme des carrés de a et b = 25
c = 3.5
```

- Paramètres formels: **x**,**y**
  Paramètres effectifs: **a**,**b**

Exercice 2.

- Appels erronés:

(1): deuxième paramètre (b) de type réel. (*Il doit être de type entier*)
(3): l'appel d'une fonction ne doit pas constituer, tout seul, une instruction.
(8): deuxième paramètre (2.5) est une constante de type réel. (*Il doit être de type entier*)
(10): la valeur du deuxième paramètre (a/b) est de type réel. (*Il doit être de type entier*)
* Expliquer aux étudiants que les autres appels sont vrais.

Exercice 3.

```
Algorithme exo3
Var a,b:entier
Fonction min (x,y:entier):entier
Début
  Si x<=y Alors
    min ← x
  Sinon
    min ← y
  Finsi
FinFonction
Début      {Algorithme principal}
 Ecrire("a = ")
 Lire(a)
 Ecrire("b = ")
 Lire(b)
 Ecrire("Le minimum de a et b = ",min(a,b))
Fin
```

Exercice 4.

```
Program exo4;
Uses crt;
var pbase,gbase,haut: real;
function strapeze(Pb, Gb, H:real):real;
begin
  strapeze:=(Pb+Gb)*H/2;
end;
begin  {programme principal}
 clrscr;
 write('Petite base = ');
 readln(pbase);
 write('Grande base = ');
 readln(gbase);
 write('Hauteur = ');
 readln(haut);
 writeln('La surface = ',strapeze(pbase, gbase, haut));
 readkey;
end.
```

Exercice 5.

```
Program exo5;
Uses crt;
var a,b: integer;
function puissance(x, n:integer):integer;
var i,p:integer;
begin
  p:=1;
  For i:=1 To n Do
    p:=p*x;
  puissance:=p;
end;
begin {programme principal}
 clrscr;
 write('a = ');
 readln(a);
 write('b = ');
 readln(b);
 writeln('a^b = ',puissance(a,b));
 writeln('b^a = ',puissance(b,a));
 readkey;
end.
```

Exercice 6.

```
Program exo6;
Uses crt;
function pgcd(x, y:integer):integer;
begin
  while (x*y <> 0) Do
  begin
    if x>y then
      x:=x-y
    else
      y:=y-x;
  end;

  if x=0 then
     pgcd:=y
  else
     pgcd:=x;
end;

begin {programme principal}
 clrscr;
 writeln('PGCD de 114 et 36 = ',pgcd(114,36));
 writeln('PGCD de 230 et 60 = ',pgcd(230,60));
 readkey;
end.
```

Exercice 7.

```
Algorithme exo7
Var Q,P,C:entier
Fonction fact(n:entier):entier
Var i,f:entier
Début
  f ← 1
  Pour i=1 à n faire
    f ← f*i
  FinPour
  fact ← f
FinFonction
Début          {Algorithme principal}
 Ecrire ("Q = ")
 Lire(Q)
 Ecrire ("P = ")
 Lire(P)
 Ecrire ("(Q!(Q-P)!)/P! = ",fact(Q)*fact(Q-P)/fact(P))
Fin
```

Exercice 8.

a = 2 b = 4

c = a*a + b*b = 4+16 = 20

d = somme(a,1)* somme(a,1) + b*b

= (a+1)*(a+1) + b*b

= 9+16 = 25

Exercice 9.

1.

(1) `Ecrire(multiplier(a,b,c))` ➔ ***incorrecte****, l'appel d'une procédure doit constituer, tout seul, une instruction.*

(2) `multiplier(a,b,c)` ➔ ***correcte***

(3) `multiplier(a,b,a)` ➔ ***correcte***

(4) `c ← multiplier(a,b,c)` ➔ ***incorrecte****, l'appel d'une procédure doit constituer, tout seul, une instruction.*

(5) `multiplier(a,b)` ➔ ***incorrecte****, il manque un paramètre.*

(6) `multiplier(a,b,d)` ➔ ***incorrecte****, le paramètre d est entier (il doit être réel).*

2.

L'instruction `multiplier(4.0,3.0, c)` est ***juste***. Elle calcule :

c = 4.0*3.0

Exercice 10.

1.

Variables locales de la procédure *proc1*: **x1**, **y1**

Variables locales de la procédure *proc2*: **x2**, **y2**

Variables globales: **a**, **b**, **c**

2.

(1) `a ←x*y` ➔ ***autorisée***

(2) `Ecrire(x1)` ➔ ***autorisée***

(3) `b ←x*y2` ➔ ***non autorisée***, y2 est une variable locale de la procédure *proc2*.

(4) `x2 ← a*b` ➔ ***autorisée***

(5) `y2 ← y1` ➔ ***non autorisée***, y1 est une variable locale de la procédure *proc1*.

(6) `Ecrire(y)` ➔ ***non autorisée***, y est un paramètre de la procédure *proc1*.

(7) `Lire (a,b)` ➔ ***autorisée***

(8) `Lire (x1,x2)` ➔ ***non autorisée***, x1et x2 sont des variables locales.

(9) `c ← z*x` ➔ ***non autorisée***, z et x sont des paramètres formels.

Exercice 11.

1. Les valeurs de **a** et **b** n'ont pas changé après l'appel de la procédure *echanger*.

2. La valeur de **a** a changé mais pas celle de **b,** après l'appel de la procédure *echanger*.

3. Les valeurs de **a** et **b** ont changé tous les deux après l'appel de la procédure *echanger*.

4. Les paramètres **x** et **y** sont des paramètres en **entrée/sortie** parce qu'on en a besoin en entrée et en sortie de la procédure. **Conclusion:** les paramètres en **sortie** ou en **entrée/sortie** doivent être passés par adresse (précédés par le mot-clé **Var**).

5. Il y a une erreur. **Conclusion:** les paramètres effectifs qui correspondent à des paramètres passés par adresse *ne peuvent pas être des constantes.*

Exercice 12.

```
program exo12;
uses crt;

function premier(e:integer):boolean;
var i,m:integer;
    ok:boolean;
begin
  ok:=true;
  i:=2;
  m:= e div 2;
  while (i<=m) and (ok=true)  do
  begin
    if (e mod i)=0 then
      ok:=false
    else
      i:=i+1;
  end;
  premier:=ok;
end;

procedure nombre_pre(N:integer);
var i:integer;
begin
 for i:=1 to N-1 do
  if (premier(i)=true) then
    write(i,'  ');
end;

begin
  clrscr;
  nombre_pre(1000);
  readkey;
end.
```

Exercice 13.

```
program exo13;
uses crt;
const N=3;
      M=3;
type Mat = array[1..N,1..M] of integer;
var X,Y,Z,P,Xt,Yt,T:Mat;
procedure lire_mat(var A:Mat);
var i,j:integer;
begin
 for i:=1 to N do
   for j:=1 to M do
     read(A[i,j]);
end;
```

```
procedure aff_mat(A:Mat);
var i,j:integer;
begin
 for i:=1 to N do
   for j:=1 to M do
     write(A[i,j],' ');
end;

procedure somme_mat(A,B: Mat; var S:Mat);
var i,j:integer;
begin
 for i:=1 to N do
   for j:=1 to M do
     S[i,j]:=A[i,j]+B[i,j];
end;

procedure prod_mat(A,B: Mat; var C:Mat);
var i,j,k:integer;
begin
 for i:=1 to N do
   for j:=1 to M do
    begin
     C[i,j]:=0;
     for k:=1 to M do
        C[i,j]:=C[i,j]+A[i,k]*B[k,j];
    end;
end;

procedure trans_mat(A:Mat;var At:Mat);
var i,j:integer;
begin
 for i:=1 to N do
   for j:=1 to M do
     At[i,j]:=A[j,i];
end;

begin
 clrscr;
 writeln('Entrer les elements de la matrice X: ');
 lire_mat(X);
 writeln('Entrer les elements de la matrice Y: ');
 lire_mat(Y);
 somme_mat(X,Y,Z);
 writeln('La somme de X et Y: ');
 aff_mat(Z);
 prod_mat(X,Y,P);
 writeln('Le produit de X et Y: ');
 aff_mat(P);
 trans_mat(X,Xt);
```

```
 writeln('Le transpose de X: ');
 aff_mat(Xt);
 trans_mat(Y,Yt);
 writeln('Le transpose de Y: ');
 aff_mat(Yt);
 somme_mat(Xt,Yt,T);
 writeln('La somme de Xt et Yt: ');
 aff_mat(T);
 readkey;
end.
```