Co-Exercice 1.1 :

1. La deuxième expression « taper des réels a et b » est à côté du premier expression « bonjour » (collé, sans espace !)
2. La commande writeln fait passer le curseur à la ligne : les deux expressions sont maintenant l'un en-dessous de l'autre.
3. si b≥0 Turbo Pascal fournit une valeur approchée du nombre $(a+\frac{\sqrt{b}}{2})$ et non l'expression exacte (et formelle).
Si b<0 Turbo Pascal déclaré une erreur d'exécution.
4. on remplace : write('taper des réels a et b');
read(a,b);
par : write('Entrer un réel a') ; readln(a) ;
write('Entrer un réel b'); readln(b);
5. le programme fournit deux nombres différents, par exemple si on donne a=1 et b=4 :
pour la premier version x=$1+\frac{\sqrt{4}}{2}$=2
pour la deuxième version x=$\frac{1+\sqrt{4}}{2}=\frac{1+2}{2}=1.5$

CO-Exercice 1.2 :

La manière la plus simple d'arriver au même résultat est : write('x='); readln(y); puis write('y=');readln(x).
Sinon, une piste est d'écrire : x:=y; y:=x;. Le problème est que lors de la première affectation l'ancienne valeur du x est écrasé, donc ce qu'on met dans le y c'est la nouvelle valeur de x ... à savoir y !
Il faut donc stocker la valeur de x avant de l'écraser : on obtient la syntaxe temp:=x ; x:=y ; y:=temp ;.
Autre méthode pour avoir la même action :

```
PROGRAM Permutation;
VAR x,y: real;
BEGIN
write('x='); readln(x);
write('y='); readln(y);
x:=x+y; y:=x-y ;
x:=x-y;
writeln('après traitement : '); writeln('x=',x, ',y=',y);
end.
```

Co-Exercice 1.3 :

```
1. PROGRAM naissance;
VAR x:INTEGER;
BEGIN
write('Entrez votre année de naissance'); readln(x);
write('Vous avez ',2016-x, 'ans');
END.
```

```
2. PROGRAM température ;
VAR F:real;
BEGIN
write('Entrer la température en degrés Farenheit');
readln(F);
writeln('La température en degrés Celsius est ',
(F-32)/18);
END.
```

```
3. PROGRAM temps;
VAR T,s,m,m1,h : integer;
BEGIN
write('Entrer le temps exprime en secondes :');
readln(T);
s:=T mod 60;
m1 := T div 60;
{ On a transformé notre temps T en minutes et
secondes ; c'est donc ce nombre m1 de minutes qu'il
faut maintenant transformer en heures et minutes }
m:= m1 mod 60 ;
h:= m1 div 60 ;
writeln('Le temps est de ',h, ' heures ', m, ' minutes et
', s, ' secondes.');
END.
```

Autre rédaction possible : T=h*3600+m*60+s d'où
h:=T div 3600;
m:=(T mod 3600) div 60 ;
s:=(T mod 3600) mod 60;
Sur un exemple, convertir 11843 secondes :
11843= 3*3600+1043
le 1 correspond à T DIV 3600 : nombre d'heures.
Le reste (=1043), soit T MOD 3600, est encore exprimé en secondes : pour obtenir les minutes, 1043 DIV 60 =17, et pour obtenir le reste du reste, 1043 MOD 60 soit 23 secondes.
Finalement, 11843 secondes = 3h 17 min 23 s .

Co-Exercices 2.1 :

```
Algorithme calcul
Variable a,b : entier
Début
Ecrire (' a : ')   Lire(a)
Ecrire (' b : ')   Lire(b)
Ecrire('a+b= ',a+b)
Ecrire('a-b=',a-b)
Ecrire ('a*b= ',a*b)
Si (b≠0) alors Ecrire('a/b= ',a/b)
Sinon Ecrire('On ne peut pas diviser par 0')
Fsi
END
```

Co-Exercices 2.2 :

Si l'utilisateur saisi un nombre positif non nul ,le programme affiche la valeur de cette expression $\sqrt{4*a}-2$ (ex : a=9 la valeur affiché= 4)
Si l'utilisateur saisi un nombre négatif ou nul, le programme affiche la variable a sans modification.
Pour gagner une ligne de code on remplace (a :=4*a ; a :=sqrt(a) -2 ;) par a :=sqrt(4*a)-2 ;

Co-Exercice 2.3 :

```
PROGRAM Equation1 ;
VAR a,b: REAL ;
BEGIN
writeln('On veut résoudre a*x+b=0');
```

```
write('Donnez  a et b  :');
readln(a,b);
IF (a=0) THEN
  IF(b=0) THEN writeln('Tout nombre est solution')
  ELSE writeln('Aucune solution')
ELSE writeln('La solution est ', -b/a :2 :2) ;
Readln ;
End.
```

Co-Exercice 2.4

```
Program  maxabc ;
Var a,b,c,max :real ;
Begin
Writeln('Entrez les nombres a,b,c') ; Readln(a,b,c) ;
Max :=a ;
If  max<b then max :=b ;
If  max<c then max :=c ;
Writeln('max= ',max :2 :2) ;
Readln ;
End.
```

Co-Exercice 2.5 :

```
PROGRAM Equation2 ;
VAR a,b,c, delta : REAL ;
BEGIN
writeln('On veut résoudre a*x^2+b*x+c=0');
write('Donnez a,b et c :');
readln(a,b,c);
IF (a=0) THEN                {l'équ est du 1er degré }
 IF (b=0) THEN                { a=b=0 }
  IF (c=0) THEN
    writeln('Tout nombre est solution') {a=b=c=0}
  ELSE writeln('Aucune solution') { a=b=0, c≠0 }
 ELSE writeln('La solution est ', -c/b) {a=0 et b≠0}
ELSE BEGIN delta:=SQR(b)-4*a*c; { a≠0 }
 IF (delta=0) THEN    {Δ= 0}
 writeln('La solution double est : ',-b/(2*a))
 ELSE  IF (delta>0) THEN    { Δ> 0}
  writeln('Les deux solutions réelles sont :',
(-b-SQRT(delta))/(2*a),'et ',(-b+SQRT(delta))/(2*a))
        ELSE      {Δ < 0}
  writeln('La résolution est hors-programme');
END;
END.
```

Co.exercice 3.1

Algorithme0 :	Algorithme1 :	Algorithme4 :
bonjour2	bonjour	bonjour
bonjour3	fin	bonjour
bonjour4	Algorithme2 :	fin
bonjour5	rien afficher	Algorithme5
bonjour6	Algorithme3 :	0
bonjour7	bonjour	1
bonjour8	bonjour	2
fin	bonjour	fin
	fin	

Co.exercice 3.2

```
Variables N, i :Entier
Debut
Ecrire ('Entrez un nombre : ')
Lire (N)
Ecrire('La table de multiplication de ce nombre est
:')
Pour i= 1 à 10
  Ecrire (N, ' x ', i, ' = ', n*i)
Finpour
Fin
```

Co-exercice 3.3

Suite mise en jeu : $u_0 = 0$ et $\forall n \in \mathbb{N}$, $u_{n+1} = 2u_n + 3$

i	instruction	Valeur de u
0	/	U=0
1	U=2*0+3	U=3
2	U=2*3+3	U=9
3	U=2*9+3	U=21
4	U=2*21+3	U=45
5	U=2*45+3	U=93
6	U=2*93+3	U=189

Pour que le programme n'affiche que u_{90}, il suffit de calculer les termes de la suite jusqu'à u_{90} :**FOR** i:=1 TO 90, et de sortir le **write**(u) de la boucle FOR, afin qu'il n'y ait pas d'affichage à chaque étape.

```
Program suite ;
Var u :real ; i :integer ;
Begin
u :=0 ;  i :=1 ;  writeln(u) ;
while (i<=6) do
 begin u :=2*u+3 ;   writeln(u) ;
        i :=i+1 ; end ;
end.
```

Co-exercice 3.4

```
PROGRAM puissance;
VAR  x,p : REAL ; n,k : INTEGER ;
BEGIN
WRITE('Donner x et n'); READLN(x,n);
p:=1;
IF (n>=0) THEN
   FOR k:=1 TO n DO
      p:=p*x
ELSE
  IF x<>0 THEN
     FOR k:=1 TO (-n) DO  p:=p/x;

IF (x=0) and (n<0) THEN
        WRITELN('Erreur division par 0') ;
ELSE WRITELN(x:1:2,' ^',n,'=',p:1:2);
END.
```

Co-exercice 3.5

```
PROGRAM factorielle;            ,
VAR n, k, fact : INTEGER;
BEGIN
WRITELN('Entrer un entier n positif') ;
READLN(n);.
IF (n < 0) THEN  WRITELN('La valeur de n est mal
choisie')
ELSE BEGIN
       fact:= 1 ;
      FOR k:=1 TO n DO    fact:=fact*k;
 WRITELN(n, '!= ' , fact);   end ;
END.
```

Problème et solution : En fait la case mémoire réservée à une variable de type integer est trop petite pour des entiers au-dessus de 32767, donc le programme précédent ne donne pas le bon résultat dès n=8. Pour résoudre le problème, on peut considérer le calcul dans les réels et non dans les entiers, en déclarant fact :real;(ou fact :longint ;).

Co-exercice 4.1

1 la première version

```
 Algorithme Plusgrant
Variables N, i, PG : Entier
Debut
 PG ← 0                                  ..... 4
 Pour i ← 1 à 5
  Ecrire ('Entrez un nombre : ')
  Lire (N)
  Si (i = 1) ou (N > PG) Alors       ..... 8
    PG ← N
  FinSi
 FinPour
 Ecrire ('Le nombre le plus grand était : ', PG)
Fin
```

En ligne **4**, on peut mettre n'importe quoi dans PG, il suffit que cette variable soit affectée pour que le premier passage en ligne **8** ne provoque pas d'erreur.

2 Pour la version améliorée, cela donne :

```
Variables N, i, PG, IPG en Entier
Debut
 PG ← 0
 Pour i ← 1 à 5
    Ecrire ('Entrez un nombre : )
    Lire (N)
    Si (i = 1) ou (N > PG) Alors
       PG ← N
       IPG ← i
    FinSi
 FinPour
 Ecrire ('Le nombre le plus grand était : ', PG)
 Ecrire ('Il a été saisi en position numéro ', IPG)
Fin
```

3 Pour la version le plus générale

```
Variables N, i, PG, IPG : Entier
Debut
 N ← 1
 i ←0
 PG ← 0
 TantQue N <> 0 faire
   Ecrire ('Entrez un nombre : ')
   Lire (N)
   i ← i + 1
 Si( i = 1) ou ( N > PG)  Alors
      PG ← N
      IPG ← i
   FinSi
 FinTantQue
 Ecrire ('Le nombre le plus grand était : ', PG)
 Ecrire ('Il a été saisi en position numéro ', IPG)
Fin
```

Co-exercice 4.2

1. Affichage **vertical** : 1 2 3 4 5 6 (car **writeln** et non **write**)

2.
```
REPEAT x:=x+1; WRITELN(x);
  UNTIL x>5 ;
```

Représentation de l'action de ces boucles par un tableau :

WHILE : on teste avant de modifier x

relation x ≤ 5		V	V	V	V	V	F : arrêt
X	0	1	2	...	5	6	

REPEAT UNTIL : on modifie x puis on teste

X	0	1	2	...	5	6
relation x > 5		F	F	...	F	V : arrêt

For : l'instruction x :=0 ; est inutile.

```
For x :=1 To 5 Do
     WRITELN(x) ;
```

Co-exercice 4.3

1.
```
BEGIN
randomize;
rlt:=random(5);
REPEAT
WRITE('Entrez votre proposition');
READLN(k);
UNTIL k=rlt;
WRITELN('Vous avez gagné');                ;
End.
```

2.
```
BEGIN
randomize;
rlt:=random(5);
c:=0;
REPEAT
WRITE('Entrez votre proposition'); READLN(k);
c:=c+1;
```

```
UNTIL k=rlt ;
 WRITELN(' Vous avez gagné en ', c,'essais');
end.
```

3. Pour mettre une boucle **while**, commencer par introduire k avant de répéter ! :

```
BEGIN
randomize;
rlt:=random(5);
WRITE('Entrez votre proposition'); READLN(k);
WHILE (k<>rlt) ;
BEGIN
WRITE('entrez votre nouvelle proposition');
READLN(k);
END;
WRITELN('gagné !');
end.
```

Co-exercice 4.4

```
PROGRAM jeubis;
VAR  x,y:REAL;   compt : INTEGER;
BEGIN
x:=random(65);
WRITE('Entrez votre proposition');
READLN(y);
compt:=1;
WHILE x<>y DO
  BEGIN
   IF x< y then WRITE('trop grand')
   ELSE   WRITE('trop petit');
  WRITE('Entrer votre nouvelle proposition');
READLN(y);
   compt:=compt +1;
  END;
IF compt <= 5 then
WRITE(' Bravo ! Vous avez gagné en ', compt,
'essais')
ELSE IF compt <= 10 then
WRITE(' c''est bien ! Vous avez gagné en ', compt,
'essais')
ELSE WRITE('changez de méthode car il vous a
fallu', compt, 'essais');
END.
```

Co-exercice 4.5

```
program calpi;
var  pi,e:real;
      x,j:integer;
begin
write('Entrez un nombre epsilon: ');
readln(e);
x:=1; j:=1;pi:=0;
while (1/x)>=e do
begin pi:=pi+j*(1/x);
          x:=x+2;     j:= -1*j;
end;
write('pi= ',4*pi:2:5);
readln;
end.
```

Co-exercice 4.6

```
program PGCD ;
var a,b,r :integer ;
begin
writeln('POUR CALCULER LE PGCD DE DEUX
NOMBRES') ;
write('veuillez saisir le premier nombre s'il vous
plait') ;
readln(a) ;
write('veuillez saisir le deuxième nombre s'il vous
plait');
readln(b) ;
while b<>0 do
begin r := a mod b ;
          a :=b ;  b :=r ;
end ;
writeln('Le PGCD de ces deux nombres est: ',a) ;
end.
```

```
Program PGCD ;
Var a,b :integer ;
Begin
Writeln('POUR CALCULER LE PGCD DE DEUX
NOMBRES') ;
Write('veuillez saisir le premier nombre s'il vous
plait ') ; readln(a) ;
Write('veuillez saisir le deuxième nombre s'il vous
plait');readln(b) ;
While a<>b do
Begin
If a>b then a :=a-b
Else b :=b-a :
End ;
writeln('Le PGCD de ces deux nombres est: ',b) ;
end.
```

```
Program PPCM ;
Var a,b,c,d :integer ;
Begin
Writeln('POUR CALCULER LE PPCM DE DEUX
NOMBRES') ;
Write('veuillez saisir le premier nombre s'il vous
plait ') ; readln(a) ;
Write('veuillez saisir le deuxième nombre s'il vous
plait');readln(b) ;
C :=a ; d :=b ;
While a<>b do
Begin
If a>b then b :=d+b ;
Else a :=c+a :
End ;
writeln('Le PPCM de ces deux nombres est: ',a) ;
end.
```

Co-exercice 4.7

```
program Nbpremiers;
uses crt;
var  nombre, i,  j, N: integer;
begin
clr scr;
WRITE('Entrer un entier N positif'); READLN(N);
nombre := 1;
j :=1 ;
while j<= N do
begin
 i := 2;
 while (((nombre mod i) <> 0) and (i<=
(trunc(nombre div 2) + 1))) do
    i := i + 1;

 if((nombre mod i) <> 0) OR   (nombre=2) OR
(nombre=3) then
   begin
   writeln('Element  ',j,'vaut  ',nombre);
   j := j + 1;
   end;
 nombre := nombre + 1;
end;
readln;
end.
```

i<=(Trunc(nombre div 2)+1)
Cette condition permet de ne chercher que parmi les nombres compris entre 2 et la moitié du nombre en question augmenté de 2 (en effet, à chaque itération de la boucle principale, *i* commence à 2 et le reste de la division d'un nombre par un autre supérieur à sa moitié sera toujours supérieur à 0).

nombre mod i<>0
Cette condition assure que le nombre ne doit pas être divisible par les différentes valeurs de *i.*

Co-Exercice 5.1
Cet algorithme déclare et remplit le tableau Truc de 7 valeurs numériques,et en les mettant toutes à zéro.

Co-Exercice 5.2
Algorithme 1 remplit un tableau avec six valeurs : 0, 1, 4, 9, 16, 25.
Il les écrit ensuite à l'écran. Simplification

```
Début
Pour i ← 0 à 5
  Nb(i) ← i * i
  Ecrire (Nb(i))
Finpour
Fin
```

Algorithme 2 remplit un tableau avec les sept valeurs : 1, 3, 5, 7, 9, 11, 13.
Il les écrit ensuite à l'écran. Simplification

```
Début
N(0) ← 1
Ecrire N(0)
Pour i ← 1 à 6
  N(i) ← N(i-1) + 2
  Ecrire N(i)
Finpour
Fin
```

Algorithme 3 remplit un tableau de 8 valeurs : 1, 1, 2, 3, 5, 8, 13, 21

Co-Exercice 5.3

```
Algorithme MoyNote
Tableau Notes(6) : Entier
Variable i :Entier
Début
s ← 0
Pour i ← 0 à 6
  Ecrire ('Entrez la note numéro ', i + 1)
  Lire (Notes(i))
  s ← s + Notes(i)
Finpour
Ecrire ('Moyenne :', s/7)
Fin
```

Co-Exercice 5.4

```
 program tableaux ;
var  tab : array [ 1 . . 1 0 ] of  integer ;
      max, min, s, i : integer ;
begin
randomize ;
for i :=1 to 10 do
tab[ i ] :=random(26)+15 ;
for i:=1 to 10 do
wr iteln (tab[ i ]) ;
max :=tab[1] ;  min :=tab[1] ;
for i :=2 to 10 do
  if  max< tab[i] then
      max := tab[i] ;
for i :=2 to 10 Do
   if min> tab[i] then
      min :=tab[i] ;
s :=0 ;
for i :=1 to 10 do
s :=s+tab[i] ;
write('max= ',max,' min= ',min,' moy= ',s/10 :2 :2) ;
readln ;
end .
```

Co-Exercice 5.5

```
program décomposition;
uses crt;
var
  tableau: array[1..100] of integer;
  i, j, nbre, Nbelm, dv: integer;
```

```
begin
 clrscr;
 NbElm := 0;{nombre élément(facteur)}
 dv := 2;  {diviseur}
 write('Entrez le nombre dont vous voulez la
décomposition : ');
 readln(nbre);
if nbre>0 then
begin
 while (nbre <> 1) do
 begin
  if ((nbre mod dv) = 0) then
  begin
   NbElm := NbElm + 1;
   tableau[NbElm] := dv;
   nbre := nbre div dv;
  end
  else
    dv := dv + 1;
 end;
 i := 1;
 while (i < NbElm) do
 begin
  write(tableau[i], '*');
  i := i + 1;
 end;
 writeln(tableau[i]);
end ;
 readln;
end.
```

Co-Exercice 5.6

```
program tableaux_db ;
var  tab : array [ 1..4,1..3 ] of  integer ;
      imax,jmax,max, i, j : integer ;
begin
randomize ;
for i :=1 to 4 do
  for j :=1 to 3 do
      tab[ i,j ] :=random(10) ;
for i:=1 to 4 do
  begin for j :=1 to 3 do
             write (tab[ i,j ]) ;
          writeln ;
  end ;
imax :=1 ;  jmax :=1 ;
for i :=1 to 4 do
 for j := 1 to 3 do
   if  tab[imax,jmax]< tab[i,j] then
     begin
        imax := i ;  jmax :=j ;
     end ;
write('Le plus grand élément est :  ',tab[imax,jmax]) ;
writeln('Il se trouve aux indices:  ',imax, ' ; ',jmax) ;
readln ;
end.
```

Co-Exercice 5.7

```
program sommat ;
var  som,A,B : array [ 1..4,1..3 ] of  integer ;
      i, j : integer ;
begin
randomize ;
for i :=1 to 4 do
  for j :=1 to 3 do
   begin        { remplissage les matrices A, B}
     A[ i,j ] :=random(10) ;
     B[ i,j ] :=random(10) ;
   end ;
writeln('Mat A :') ; {afficher A}
for i:=1 to 4 do
  begin for j :=1 to 3 do
             write (A[ i,j ],'  ') ;
          writeln ;
  end ;
writeln('Mat B :') ; {afficher B}
for i:=1 to 4 do
  begin for j :=1 to 3 do
             write (B[ i,j ],'  ') ;
          writeln ;
  end ;

for i:=1 to 4 do   {l'addition de A et B}
   for j :=1 to 3 do
       som[ i,j ] :=a[i,j]+b[i,j] ;
writeln('La somme de ces  matrices :') ;
for i:=1 to 4 do  {afficher la somme}
  begin for j :=1 to 3 do
             write (som[ i,j ],'  ') ;
          writeln ;
  end ;
readln ;
end.
```