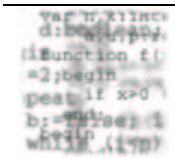

TurboPascal - Prépa HEC Ipecom
Corrections - Exemples - Exercices
vendredi 05 avril

1 Corrections des exercices du 22 mars

1.1

Ecrire un programme qui saisit deux entiers et affiche le plus grand.

```
program lemax1;
var n1,n2:integer;
begin
  writeln('Deux entiers please...'); readln(n1,n2);
  if n1>n2 then writeln(n1) else writeln(n2);
end.
```

ou

```
program lemax2;
var n1,n2:integer;
begin
  writeln('Deux entiers please...'); readln(n1,n2);
  writeln((n1+n2)/2+abs((n1-n2)/2));
end.
```

Ecrire un programme qui saisit trois entiers et affiche le plus petit.

```
program lemin;
var n1,n2,n3:integer;
begin
  writeln('Trois entiers please...'); readln(n1,n2,n3);
  if n1<n2
    then if n1<n3 then writeln(n1) else writeln(n3)
    else writeln(n2);
end.
```

1.2

Ecrire un programme qui saisit trois entiers et les permute trois à trois [il y a bien sur trois permutations possibles].

```
program perm;
var n,n1,n2,n3:integer;
begin
  writeln('Trois entiers please...'); readln(n1,n2,n3);
  for i:=1 to 3 do
    begin
      n:=n1; n1:=n2; n2:=n3; n3:=n;
      writeln('n1 : ',n1,', n2 : ',n2,', n3 : ',n3);
    end;
end.
```

1.3

Ecrire un programme qui saisit deux nombres entiers a et b et affiche a^b [par une boucle]. Même chose avec des réels, sans boucle et en utilisant `exp()` et `log`. [la correction utilise des fonctions, encore plus utiles...]

```
function p_i(a,b:integer):interger;
var i:integer;
begin
  p_i:=1;
  for i:=1 to b do p_i:=p_r*a;
end.

function p_r(a,b:real):real;
begin
  p_r:=exp(b*log(a));
end.
```

1.4

Ecrire un programme qui saisit les coefficients a, b et c d'un polynôme $ax^2 + bx + c$, et en donne le discriminant et les racines.

```
program poly;
var a,b,c,d,x,y:real;
begin
  write('Coefficient de degre 2 :');readln(a);
  write('Coefficient de degre 1 :');readln(d);
  write('Coefficient de degre 0 :');readln(c);
  d:=b*b-4*a*c;
  x:=-b/(2*a); y:=sqrt(abs(d))/(2*a);
  if d>0 then
    begin writeln('Deux solutions reelles.'); writeln(x+y,' et ',x-y); end;
  if d=0 then
    begin writeln('Une solution reelle double.'); writeln(x); end;
  if d<0 then
    begin writeln('Deux solutions complexes conjuguees.');
    writeln(x,'+',y,'i et ',x,'-',y,'i'); end;
end.
```

1.5

Ecrire un programme qui saisit 10 nombres entiers dans un tableau, et affiche le plus grand, le plus petit et la moyenne.

```
program tab;
var t:=array[1..10] of integer;
    i,g,p,s:integer;
begin
  for i:=1 to 10 do begin write(i,'ieme nombre entier :'); readln(t[i]); end;
  g:=t[1]; p:=t[1]; m:=t[1];
  for i:=2 to 10 do
    begin
      if t[i]>g then g:=t[i];
```

```
      if t[i]<p then p:=t[i];
      s:=s+t[i];
    end;
  writeln('Max : ',g); writeln('Min : ',p); writeln('Moy : ',(s/10));
end.
```

1.6

Ecrire un programme qui saisit des nombres entiers tant que leur somme est inférieure à 1000, puis qui affiche la somme obtenue et le nombre d'entiers saisis.

```
program saisie1;
var i,s,n:integer;
begin
  i:=0; s:=0;
  while s<1000 do
    begin
      write('un entier : '); readln(n);
      s:=s+n; i:=i+1;
    end;
  writeln('somme : ',s); writeln('nombre :',i);
end.
```

ou

```
program saisie2;
var i,s,n:integer;
begin
  i:=0; s:=0;
  repeat
    write('un entier : '); readln(n);
    s:=s+n; i:=i+1;
  until s<1000;
  writeln('somme : ',s); writeln('nombre :',i);
end.
```

1.7

A l'aide du programme donné en exemple, écrire un programme qui donne tous les nombres premiers entre 1 et 100.

```
program premiers;
const nmax=100;
var n,i:integer;
    d:boolean;
begin
 n:=2;
 repeat
   b:=false; i:=2;
   while (i<n)and(not(d)) do
     begin
     d:=((n mod i)=0);
     i:=i+1;
   end;
```

```
   if not(d) then writeln(n,' est premier');
   n:=n+1;
 until n>nmax;
end.
```

2 Corrections des exercices de concours du 29 mars

2.1 HEC 99

- ...corps de la procédure `Procedure Calcul1(Var V : vecteur);` qui doit retourner dans V le résultat de RV:

```
Procedure Calcul1(Var V : vecteur);
var i:integer;
    VV:vecteur;
  begin
    for i:=1 to N-1
      do VV[i]:=(1-beta*Delta*i*(N-i))*V[i]+(beta*Delta*(i+1)*(N-1-i))*V[i+1];
    i:=N; VV[i]:=(1-beta*Delta*i*(N-i))*V[i]; { on a en fait VV[N]:=V[N] }
    for i:=1 to N do V[i]:=VV[i];
  end;
```

- ...corps de la procédure `Procedure Calcul2(Var V : vecteur, i :  integer);` qui doit retourner dans V le contenu de W_i...

```
Procedure Calcul2(Var V : vecteur, i : integer);
var k:integer;
  begin
    for k:=1 to N-1 do V[k]:=0; V[N]:=1; for n:=1 to i do Calcul1(V);
  end;
```

2.2 HEC 99

- ...procédure `procedure Tirage(var C:Tableau);` permettant de simuler le tirage avec remise de 100 boules dans une urne contenant des boules de couleur C_1 ou C_2 ou C_3 en proportion respectivement 1/4, 1/4, 1/2...

```
procedure Tirage(var C:Tableau);
var i,r:integer;
  begin
    for i:=1 to 100 do
      begin r:=random(4); if r<2 then C[i]:=1 else C[i]:=r; end;
  end;
```

- ...fonction `Difference` de paramètre C qui retourne la valeur de $X_1 - X_2$...

```
function Difference(C:Tableau):integer;
var i,x1,x2:integer;
  begin
    x1:=0; x2:=0;
    for i:=1 to 100 do
      begin
```

```
        if C[i]=1 then x1:=x1+1;
        if C[i]=2 then x2:=x2+1;
      end;
    Difference:=x1-x2;
  end;
```

ou

```
function Difference(C:Tableau):integer;
var i:integer;
  begin
    Difference:=0;
    for i:=1 to 100 do
      Difference:=Difference+1*(C[i]=1)+(-1)*(C[i]=2);
      { 1*(C[i]=1) vaut 1 si C[i]=1 et 0 sinon }
  end;
```

2.3 HEC 99

- Que fait la procédure `X`?
  Elle remplit le tableau `T` de nombres entiers aléatoires, entre 1 *et* 20000.

- Que représentent les variables `U` et `S` à la fin du programme?
  Le programme cherche deux indices tels que `T[S]=T[i]`. Ainsi, tant que ceux-ci n'ont pas été trouvés, on prend un indice `i` croissant, et on cherche pour chaque indice `S<i` si l'égalité est observée. `U` est . . . inutile!! A la fin du programme, `U` vaut 1, *et `S` est le premier indice tel que `T[S]=T[i]`, pour `i>S`.*

- Pourquoi est-il certain que le nombre de passages dans la boucle `REPEAT ... UNTIL` est fini?
  Parce qu'il existe nécessairement deux indices tels que `T[S]=T[i]`, car on ne peut avoir 20001 *valeurs différentes dans les* 20001 *'cases' du tableau `T` remplies avec des nombres entiers compris entre* 1 *et* 20000.

2.4 ESCP01

- Que fait l'ordinateur dans le cas où les variables `a` et `b` contiennent toutes deux le même nombre?
  Si `a>b`, l'ordinateur les permute. Ensuite, si `a<b`, l'ordinateur afffiche `a` et `b`. Donc, ici, si `a=b`, l'ordinateur ne fait rien!

- Qu'affiche l'ordinateur dans le cas où les variables `a` et `b` contiennent respectivement les nombres 3 et 5?
  Si `a` et `b` contiennent 3 *et* 5, *alors on n'a pas `a>b`, donc l'ordinateur ne les permute pas. Ensuite, on a bien `a<b`, donc l'ordinateur les affiche.*

- Qu'affiche l'ordinateur dans le cas où les variables `a` et `b` contiennent respectivement les nombres 10 et 1?
  Si `a` et `b` contiennent 10 *et* 1, *on a `a>b`, et l'ordinateur les permute. `a` et `b` contiennent alors* 1 *et* 10, *on a bien `a<b`, donc l'ordinateur les affiche [il affiche les nouvelles valeurs, permutées].*

2.5 ESCP01

- – ... les valeurs successives de `alea` sont 4, 2, 3 et 2. Donner les valeurs de `A[1]`, `A[2]`, `A[3]`, `A[4]` et `A[5]`...
  La procédure affecte tout d'abord à `A` les valeurs 1, 2, 3, 4 *et* 5. *Ensuite, `i` prend des valeurs de* 5 *à* 2, *`alea` prend des valeurs aléatoires, et les valeurs de `A[i]` et `A[alea]` sont permutées. Cela donne:*
  - ∗ *permutation de `A[5]` et `A[4]`; les valeurs de `A` sont maintenant* 1, 2, 3, 5 *et* 4.
  - ∗ *permutation de `A[4]` et `A[2]`; les valeurs de `A` sont maintenant* 1, 5, 3, 2 *et* 4.
  - ∗ *permutation de `A[3]` et `A[3]`; les valeurs de `A` sont maintenant* 1, 5, 3, 2 *et* 4.
  - ∗ *permutation de `A[2]` et `A[2]`; les valeurs de `A` sont maintenant* 1, 5, 3, 2 *et* 4.
  – Quelles valeurs ... doit prendre `alea` pour obtenir ... le tableau `A[1]`= 3, `A[2]`= 5, `A[3]`= 2, `A[4]`= 4 et `A[5]`= 1?
  Au début, les valeurs de `A` sont 1, 2, 3, 4 *et* 5.
  - ∗ *On veut `A[5]`*= 1, *donc `alea`*= 1 *lorsque `i`*= 5. *`A` vaut alors* 5, 2, 3, 4 *et* 1.
  - ∗ *On veut `A[4]`*= 4, *donc `alea`*= 4 *lorsque `i`*= 4. *`A` vaut alors* 5, 2, 3, 4 *et* 1.
  - ∗ *On veut `A[3]`*= 2, *donc `alea`*= 2 *lorsque `i`*= 3. *`A` vaut alors* 5, 3, 2, 4 *et* 1.
  - ∗ *On veut `A[2]`*= 5, *donc `alea`*= 1 *lorsque `i`*= 2. *`A` vaut alors* 3, 1, 2, 4 *et* 1.
  – ... pourquoi cette procédure permet de simuler un tirage sans remise de 5 jetons ...?
  Les 5 *jetons, numérotés de* 1 *à* 5, *sont 'stockés' dans `A`. La première permutation consiste à mettre à l'indice* 5 *de `A` un jeton, tiré au hasard parmis ceux aux indices* 1 *à* 5. *La seconde consiste à mettre à l'indice* 4 *de `A` un jeton, tiré au hasard parmis ceux aux indices* 1 *à* 4, *donc ceux qui restent. La troisième met à l'indice* 3 *de `A` un jeton, tiré au hasard parmis ceux qui restent, aux indices* 1 *à* 3, *etc.*

- ... fonction d'en-tête `function T(A:tableau):integer;` qui renvoie le nombre de sous-suites croissantes du tableau `A` ...

```
function T(A:Tableau):integer;
var i,j:integer; { i:debut de la sous-suite, j:fin de la sous-suite }
    pas_fin:boolean; {pas_fin:j ne dépasse pas 5}
  begin
    T:=0;
    for i:=1 to 4 do
      begin
        pas_fin:=TRUE; j:=i+1;
        while A[j]>A[j-1] AND pas_fin do
          begin
            T:=T+1; if j<5 then j:=j+1 else pas_fin:=FALSE;
          end;
      end;
  end;
```

- Après exécution du programme la valeur affichée de S est 2.98. Ce résultat est-il étonnant?
  Cela signifie qu'il y a eu 29800 *sous-suites croissantes dans les* 10000 *permutations, soit* 2.98 *en moyenne. Etonnant?? Euh... faut voir ce qu'on a trouvé dans l'étude théorique, plus haut dans le sujet.*

3 Sujets de concours non corrigés

3.1 INSEEC93

On considère le programme suivant:

```
PROGRAM inseec;
VAR a,b,c:REAL;
    n,k,r:INTEGER;
BEGIN
  WRITE('a= '); READLN(a);
  WRITE('b= '); READLN(b);
  WRITE('donner n entier plus grand que 2. n = '); READLN(n);
  WRITE('liste complete? [0-non, 1-oui]'); READLN(r);
  FOR k:=3 TO n DO
    BEGIN
      c:=(b+a)/2;
      IF r=1 THEN WRITELN(c);
      a:=b; b:=c;
    END;
  IF r=0 THEN WRITELN(c);
END.
```

- Quelles valeurs sont écrites sur l'écran, dans les cas suivants:
  - L'utilisateur donne 1, 2, 6, 1.
  - L'utilisateur donne 1, 2, 6, 0.
- Quel est le but du programme?

3.2 ESCP95

On considère la procédure TurboPascal:

```
PROCEDURE SOM(n:INTEGER; VAR S:REAL);
VAR k:INTEGER;
  BEGIN
    S:=1/n;
    FOR k:=n-1 DOWNTO 1 DO S:=S/2+1/k;
    S:=S/2;
    WRITELN(S);
  END;
```

Justifier ce qui s'affiche sur l'écran après l'appel de `SOM(5,S)` dans un programme [à construire] contenant cette procédure.

3.3 EDHEC97

On considère la déclaration de la fonction suivante, rédigée en TurboPascal:

```
FUNCTION f(p,q:INTEGER):REAL;
VAR j:INTEGER;
    z:REAL;
  BEGIN
    IF (p<=0) OR (q<0)
      THEN WRITE('valeurs incorrectes')
      ELSE
        IF (q=0) OR (q=1)
          THEN f:=1;
          ELSE
            BEGIN
              z:=1;
              FOR j:=1 TO (q-1) DO z:=z*(1-j/p);
              f:=z;
            END;
  END;
```

- Montrer que, si p est un entier naturel non nul et q un entier naturel, $f(p,q) = \frac{A_p^q}{p^q}$.
- Utiliser cette déclaration pour écrire un algorithme en TurboPascal donnant la valeur de $E(N) = \sum_{k=0}^{n} \frac{A_n^k}{n^k}$.

3.4 EDHEC96

- * Etudier la fontion f définie sur $\mathbb{R}_+^*$ par $f(x) = x - \ln(x)$.
  * Etudier le signe de $f(x) - x$.
- On considère l'algorithme (en TurboPascal) suivant:

```
program iter;
var n,k:integer;
    a,u,p:real;
function f(x:real):real;
  begin
    if x>0 then f:=x-ln(x);
  end;
begin
  readln(n,a); u:=a; p:=a;
  for k:=1 to n do
    begin
      u:=f(u);
      p:=p*u;
    end;
  writeln(u,p);
end.
```

  * Dans le cas particulier où $n = 2$ et $a = 3$, donner les valeurs approchées à 10^{-4} près par défaut des contenus des variables u et p à la fin de l'algorithme.
    Dorénavant, dans le cas général, on note u_n et p_n les contenus respectifs des variables u et p à la fin de l'algorithme, lorsque leur calcul est possible.
  * Pour quelles valeurs de a peut-on définir la suite $(u_n)_{n\in\mathbb{N}}$ dont le premier terme est $u_0 = a$ et dont le terme général u_n est calculé par l'algorithme précédent? Pour les valeurs de a trouvées ci-dessus, donner en fonction de n le nombre

d'appels de fonctions utilisés qu cours de cet algorithme, ainsi que le nombre de soustractions, de multiplications et d'affectations nécessaires au calcul de u et p (`for k:=1 to n` n'est pas considéré comme une affectation, mais chaque appel de fonction nécessite une affectation et une soustraction)?

* Ecrire pour tout entier naturel n, la relation liant u_{n+1} et u_n.
* Pour quelle valeur de a la suite (u_n) est-elle constante?
* On suppose dans cette question $a > 1$. Montrer que la suite (u_n) est convergente et donner sa limite.
* On suppose, maintenant, $0 < a < 1$. Montrer que $u_1 > 1$. Que peut-on conclure pour la suite (u_n).

– En considérant $\ln(p_n)$, exprimer p_n en fonction seulement de a et u_{n+1}, puis la limite de p_n. Ecrire alors un nouvel algorithme en TurboPascal, ne contenant aucune multiplication et calculant p_n.

3.5 ESSEC97

On donne pour tout entier $k \geq 1$, $P(N \leq k) = \left(1 - \left(\frac{5}{6}\right)^k\right)^n$.

– Montrer que pour tout réel x, $1 - e^{-x} \leq x$.

– En déduire que l'inégalité $P(N \leq k) \geq 0.5$ implique l'inégalité: $k \geq \frac{\ln(n) - \ln(\ln(2))}{\ln(6) - \ln(5)}$.

– Ecrire un algorithme [en TurboPascal] permettant le calcul successif des 30 premiers termes de la suite $(P(N \leq k))_{k \geq 1}$ lorsque l'entier n est donné.

3.6 ESSECXX

On pose pour tout entier $j \geq 1$, $F(j) = 1 + 3.(1/3)^j - 3.(2/3)^j$.

– Etudier le sens de variation de la suite $(F(j))$. Préciser $F(1)$ et la limite de $F(j)$ lorsque j tend vers $+\infty$.

– Prouver que pour tout $y \in]0; 1[$, il existe un unique entier naturel $j \geq 2$ tel que $F(j-1) < y \leq F(j)$.

– On considère l'algorithme suivant (dans lequel y est une variable de type réel contenant une valeur appartenant à $]0; 1[$, z une variable réel, et n, p, q des variables de type entier):

```
begin
  n:=1; p:=2; q:=3; z:=(1-y)/3;
  while (p-1)/q>z do
    begin
      n:=n+1;
      p:=2*p;
      q:=3*q;
    end;
  write(n);
end;
```

Exprimer en fonction de k les valeurs contenues dans n, p, q à l'issue du k^{ieme} passage dans la boucle `while ...do`, dans la mesure où celui-ci a lieu [pour $k = 0$, il n'y a aucun passage dans la boucle et les valeurs contenues dans n, p, q sont 1, 2 et 3].

- On convient de noter j la valeur finale contenue dans la variable n.
  Exprimer en fonction de j l'avant-drnière et la dernière valeur prises par $(p-1)/q$ au cours de l'algorithme. Quelles inégalités vérifient-elles?
  En déduire que $F(j-1) < y \leq F(j)$
- Détermminer un équivalent de l'entier j tel que $F(j-1) < y \leq F(j)$, lorsque y tend vers 1.