Concours Mathématiques et Physique, Physique et Chimie, Technologie et Biologie et Géologie

Correction de l'épreuve d'Informatique

Date : Mardi 14 Juin 2005	Heure : 15 H	Durée : 2 H	Nombre de pages : 4
Barème : **Exercice 1 : 4 points**	**Exercice 2 : 4 points**	**Problème : 12 points**	

Exercice N° 1 (8.0 pts)

1) 0.5 pt

```
> S:={2*m*x+3*y-2*z=3, x+y+z=m, 5*x+4*m*y+2*z=0};
```

2-1) 0.5 pt

```
> solve(S, {x,y,z});
```

2-2) 2.0 pts = 4* 0.5 par commande

```
> with(linalg):
> A:=matrix(3,3,[[2*m,3,-2],[1,1,1],[5,4*m,2]]);
> b:=vector([3,m,0]);
> linsolve(A,b);
```

3) 0.5 pt

```
> k:=det(A);
```

4) 1.0 pt

```
> m1:=op(1,[solve(k)]); # ou encore  m1:=solve(k)[1];
> m2:=op(2,[solve(k)]); # ou encore  m2:=solve(k)[2];
```

5) 1.0 pt = 0.5 structure conditionnelle + 0.5 pour la commande inverse

```
> if m<>m1 and m<>m2  then IA:=inverse(A):
> else ERROR("L'inverse de A est non définie")
> fi;
```

6-1) 0.5 pt

```
> f:=(i,j)->i*x+j*y;
```

6-2) 1.0 pt

```
> B:=matrix(3,3,f);
```

7) 1.0 pts= 2 * 0.5 par commande

```
> AB:=evalm(A&*B);#ou encore AB:=multiply(A,B);
> evalm(A&*transpose(B)); #ou encore  multiply(A,transpose(B));
```

Exercice N2 (8.0 pts)

1) 6.0 pts

```
> restart:
> Legendre:=proc(t::symbol,n::nonnegint) (1.0=0.5 syntaxe entête+0.5 paramètres typés )
> local i, A, B ,C;                      (0.25)
> if n=0 then 1  #ou RETURN(1)           (0.5=0.25 pour cond +0.25 pour instruction)
> elif n=1 then t  #ou RETURN(t)         (0.5=0.25 pour cond +0.25 pour instruction)
> else                                   (0.5= syntaxe if ... then ... elif ... fi: )
>   A:=1: B:=t:                          (0.25)
>   for i from 2 to n do                 (1.0 = 0.5 pour syntaxe for + 0.5 pour bornes)
>     C:=((2*i-1)*t*B-(i-1)*A)/i:        (1.0)
>     A:=B:                              (0.25)
>     B:=C:                              (0.25)
>   od:
>   expand(C):  #ou RETURN(expand(C)):   (0.5)
> fi:
> end:
```

2) 2.0 pts = 0.5 pour plot + 0.5 pour {} + 0.5 pour l'appel de Legendre + 0.5pour t=-1..1

```
> plot({Legendre(t,4), Legendre(t,5),Legendre(t,6)}, t=-1..1);
```

Problème (24 pts)

Partie A

Travail demandé en algorithmique :

1) (1.5 pt)

```
FONCTION  TAILLE( ) : entier                          (0.25 pt)
VARIABLE  X : entier                                  (0.25 pt)
DEBUT
   REPETER
      ECRIRE(" Donner la dimension des tableaux")     (0.25 pt)
      LIRE(X)                                         (0.25 pt)
   JUSQU'A  (X>=1) ET (X<=NMAX)                       (0.25 pt)
   RETOURNER(X)     ( ou   TAILLE ←X)                 (0.25 pt)
FIN
```

2) (1.5 pt)

```
PROCEDURE  REMPLIR_TABLEAU (E N:entier, S T:TAB) (0.25entête)
VARIABLE  i : entier                          (0.25 pt)
DEBUT
        POUR i=1  à  N  FAIRE                 (0.25 pt)
```

```
            REPETER
                ECRIRE(" Donner l'élément du tableau d'indice " , i)      (0.25 pt)
                LIRE(T[i])                                                (0.25 pt)
            JUSQU'A  (T[i]> 0)          (0.25 pt)

        FIN_POUR
FIN
```

3)

3-1. (3.0 pts)

```
PROCEDURE  COMPTAGE ( E N : entier,  E  T : TAB,  S  C : TAB )
                                (0.25 entête+0.5 passage paramètres)
VARIABLE i, j : entier          (0.25 pt)
DEBUT
   POUR i=1  à  N  FAIRE        (0.25 pt syntaxe pour + 0.25 pt bornes)
   NBSup←0                      (0.25 pt)
      POUR j=1  à  N  FAIRE          (0.25 pt bornes)
        SI  A[j] > A[i]  ALORS  NBSup←NBSup+1  FINSI
                                (0.25 syntaxe Si ...Finsi+0.25 condition+ (0.25 instruction)
      FIN_POUR
      C[i] ←NBSup     (0.25 pt)
   FIN_POUR
FIN
```

3-2. (4.5 pts)

```
PROCEDURE  TRI_TABLEAU ( E N : entier,  E/S  T : TAB)
                                        (0.25 entête+0.25 passage paramètres)
VARIABLE      a, i,j,k : entier         (0.25)
              C:TAB                     (0.25)
DEBUT
      COMPTAGE(N, T, C)                 (0.5)
      k←0                                     (0.25)
      POUR i=1  à  N  FAIRE                   (0.25)
         j←i                                  (0.25)
         TANTQUE  j<N  ET C[j] <> k FAIRE  (0.25 TantQue+0.5 condition)
            j←j+1                             (0.25)
         FIN_TANTQUE
         a←C[i]      C[i] ←C[j]     C[j] ←a      (0.5)
         a←A[i]      A[i] ←A[j]     A[j] ←a      (0.5)
         k←k+1                                   (0.25)
      FIN_POUR
FIN
```

4) 5.0 pts

```
PROCEDURE  AFFICH_POLY (E  N : entier,  E   T,V : TAB )
                                (0.25 entête +0.25 passage paramètres)
VARIABLE  i : entier
DEBUT
   ECRIRE(A[1], "x^", B[1]);      (0.25)
```

```
POUR i=2 à N  FAIRE                    (0.25)
   SI A[i]=1 ET B[i]=1                 (0.25 condition)   + (0.25 syntaxe Si ...Finsi)
      ALORS ECRIRE ("+x")              (0.25)
      SINON  SI B[i]=1                 (0.25)
                ALORS ECRIRE ("+",A[i], "x")  (0.25)
                SINON SI A[i]=1                (0.25)
                          ALORS  ECRIRE ("+x^", B[i])            (0.25)
                          SINON  ECRIRE ("+", A[i], "x^", B[i])  (0.25)
                      FINSI
             FINSI
   FINSI
FIN_POUR
POUR i=1 à N  FAIRE                    (0.25)
   SI A[i]=1 ET B[i]=1                 (0.25 condition)
      ALORS ECRIRE ("+y")              (0.25)
      SINON  SI B[i]=1                 (0.25)
                ALORS ECRIRE ("+y^", B[i])  (0.25)
                SINON   SI A[i]=1            (0.25)
                          ALORS  ECRIRE ("+", A[i], "y")            (0.25)
                          SINON  ECRIRE ("+", A[i], "y^", B[i])     (0.25)
                      FINSI
             FINSI
   FINSI
FIN_POUR
FIN
```

5) 2.5 pts

```
ALGORITHME  DEF_POLY_ORD    (0.25)
CONSTANTE   NMAX= 20
TYPE        TAB =TABLEAU [1.. NMAX] d'ENTIER
VARIABLE    A, B : TAB      (0.25)
            N :entier       (0.25)
DEBUT
   N←TAILLE()               (0.25)
   REMPLIR_TABLEAU(N,A)     (0.25)
   REMPLIR_TABLEAU(N,B)     (0.25)
   TRI_TABLEAU(N,A)         (0.25)
   TRI_TABLEAU(N,B)         (0.25)
   ECRIRE("P=")
   AFFICH_POLY(N,A,B)       (0.25)
   ECRIRE("Q=")
   AFFICH_POLY(N,B,A)       (0.25)
FIN
```

Partie B

Travail demandé en Maple :

1) 2.0 pts

```
> COMPTAGE:=proc(A::vector, N::posint)  (0.25 entête +0.25 paramètres typés)
>    local NBSup, i,j:  (0.25)
>    global C:          (0.25)
>    C:=vector(N):      (0.25)
```

```
>    for i to N do      (0.25 boucle for)
>      NBSup:=0:
>      for j  to N do
>        if A[j] > A[i] then NBSup:=NBSup+1: fi:  (0.25 strcucture if ... fi;)
>      od:
>      C[i]:=NBSup:
>    od;
>    RETURN(eval(C));   (0.25)
> end:
```

2) 2,0 pts

```
> TRIE_TABLEAU:=proc(A::vector, N::posint) (0.25 entête +0.25 paramètres typés)
> local C, a, i, j, k:       (0.25)
> global COMPTAGE:  (0.25)
> C:=COMPTAGE(A,N);  (0.25)
> k:=0:
> for i to N do   (0.25)
>   j:=i:
>   while  j<N and C[j] <> k do   (0.25)
>       j:=j+1:
>   od:
>   a:=C[i]: C[i]:=C[j]:  C[j]:=a:   (0.25)
>   a:=A[i]: A[i]:=A[j]:  A[j]:=a:
>   k:=k+1:
> od :
> eval(A); # obligatoire si l'appel de la procédure est fonctionnel et facultatif sinon
> end:
```

3) 0.25 pt

```
> sort(P,x);
```

4) 0.25 pt

```
> f:=unapply(P,x,y);
```

5) 0.5 pt

```
> readlib(singular): singular(f(x,y));
```

6) 0.5 pt

```
> diff(f(x,y),x);
```

7) 0.5 pt

```
> (D[2]@@2)(f);
```