



## Concours Toutes Options Correction de l'épreuve d'Informatique

### Barème sur 40

#### EXERCICE 1 (8 points)

##### 1) 3 pts

- ```
1.1) > u:=unapply(sqrt(2)*cos(k*t),k); 0.25 pt  
> v:=unapply(cos(t)^k,k); 0.25 pt  
1.2) > A:=u(1); 0.25 pt  
> B:=v(1); 0.25 pt  
1.3) > eval(A, t=Pi/3); ou bien subs(t=Pi/3, A); 0.25 pt  
> eval(B, t=Pi/3); ou bien subs(t=Pi/3, B); 0.25 pt  
1.4) > sum(u(k)*v(k), k=1..n); simplify(%); 0.5 pt  
> sum(u(k), k=1..n)*sum(v(k), k=1..n); simplify(%); 0.5 pt  
1.5) > limit(u(k), k=infinity); 0.25 pt  
1.6) > limit(v(k), k=infinity); 0.25 pt
```

##### 2) 5 pts

- ```
2.1) > E1:=diff(y(t),t) + y(t) = A; 0.5 pt  
> E2:=2*diff(y(t),t$2) + diff(y(t),t)+sin(t) = B; 0.5 pt  
2.2) > dsolve({E1, y(0)=1}, y(t)); 0.5 pt  
> S1:=rhs(%); 0.5 pt  
2.3) > S2:=dsolve({E2, D(y)(0)=1, y(0)=1}, y(t), numeric); 0.5 pt  
2.4) > y:=unapply(S1, t); 0.5 pt  
2.5) > plot(y(t), t=-5..5); ou bien plot(y, -5..5); 1 pt  
2.6) > with(plots); 0.25 pt  
2.7) > odeplot(S2, [t,y(t)], t=-5..5); 0.75 pt
```

#### EXERCICE 2 (10 points)

##### 1) 2 pts

```
> Valide:=proc(Obs::list, Nb_Jour_Mois::list)  
local V;  
V:=false;  
if Obs[2]>=1 and Obs[2]<=12 and Obs[1]>=1 and Obs[1]<=Nb_Jour_Mois[Obs[2]] then  
V:=true; fi;  
return(V);  
end proc;
```

##### 2) 2 pts

```
> Num_Jour:=proc(Obs::list, Nb_Jour_Mois::list)  
local Numj, i;  
Numj:=Obs[1];  
for i to Obs[2]-1 do  
Numj:=Numj + Nb_Jour_Mois[i]; od;  
return(Numj);  
end proc;
```

```

3) 1 pt
>Avant:=proc(Obs1,Obs2::list , Nb_Jour_Mois::list)
  if Num_Jour(Obs1, Nb_Jour_Mois) < Num_Jour(Obs2, Nb_Jour_Mois) then
return(true)
else
return(false)
fi ;
end proc ;

```

#### 4) 2 pts

```

>Ajout_Obs:=proc(L ::list , Obs::list , Nb_Jour_Mois::list)
  local L1 , exist , i ;
  L1:=L;
  if Valide(Obs, Nb_Jour_Mois) then
  exist:=false ; i:=1 ;
  while exist=false and i<=nops(L1) do
    if L1[i]=Obs then exist:=true
      else i:=i+1 fi ; od ;
  if not(exist) then L1:=[op(L1),Obs] fi ; fi ;
  return(L1);
end proc ;

```

#### 5) 3 pts

```

>Tri_Obs:=proc(L ::list , Nb_Jour_Mois::list)
  local L1 , i , j , im ;
  L1:=L ;
  for i to nops(L1)-1 do
    im:=i;
    for j to nops(L1) do
      if Num_Jour(L1[j], Nb_Jour_Mois)< Num_Jour(L1[im], Nb_Jour_Mois) then
im:=j fi;od;
      x:=L1[i];
      L1[i]:=L1[im];
      L1[im]:=x;
    od;
  return(L1) ;
end proc ;

```

## PROBLEME (22 points)

### 1) 2 pts

```

Procédure Nb_Elt(bin, bsup:entier, var nb:entier)
Debut
  répéter
    lire(nb)
  jusqu'à nb>=bin ET nb<=bsup
Fin

```

### 2) 4 pts

```

Fonction Nom_Valide(Ch:chaîne de caractères) : booléen
Variable
  i:entier
  valide:booléen
Debut
  valide←faux
  Si Longueur(Ch)<=8 ET Car_Alph(Ch[1]) Alors
    i←2
    Tant que (Car_Alph(Ch[i]) OU Car_Num(Ch[i])) ET i<=Longueur(Ch) Faire
      i←i+1
    Fin Tant que
  Si i>Longueur(Ch) Alors valide←vrai Fin Si
Fin Si
Retourner(valide)

```

3) 2 pts

Procédure Saisie\_Variable(var V: chaîne de caractères, var T:entier)

Debut

  répéter

    lire(V)

  jusqu'à Nom\_Valide(V)

  répéter

    lire(T)

  jusqu'à T=1 OU T=8 OU T=16 OU T=32

Fin

4) 4 pts

Procédure Ajout\_Variable(V:chaîne de caractères, T,pos:entier, var Tnom:TAB1,  
var

Ttype,Tpos:TAB2)

Variable

  i:entier

Debut

  Si pos=1 Alors Tpos[1]←1

                  Tpos[2]←0

                  Ttype[1]←T

  Sinon

  i←1

  répéter

    i←i+1

  jusqu'à Tpos[i]=0

  Tpos[i]←pos

  Tpos[i+1]←0

  Ttype[i]←T

  Fin Si

  Pour i de 1 à Longueur(V) Faire

    Tnom[pos+i-1]←V[i]

  Fin Pour

  Tnom[pos+Longueur(V)]←''

Fin

5) 2 pts

Procédure Recup\_Nom(Tnom:TAB1, pos:entier, var TC:TABC)

Variable

  i:entier

Debut

  i←0

  répéter

    i←i+1

    TC[i]←Tnom[pos+i-1]

  jusqu'à Tnom[pos+i]='\*' OU Tnom[pos+i]='#'

Fin

6) 4 pts

Fonction Recherche(V:chaîne de caractères, Tnom:TAB1, Tpos:TAB2) : booléen

Variable

  i,j:entier

  trouve:booléen

  TC:TABC

Debut

  trouve←faux

  i←1

  Tant que trouve=faux ET Tpos[i+1]<>0 Faire

    # comparaison des longueurs

    Si Tpos[i+1]-Tpos[i]-1=Longueur(V) Alors

      Recup\_Nom(Tnom,i,TC)

```

# comparaison des 2 chaînes caractère/caractère
j←1
Tant que TC[j]=V[j] ET j<=Longeur(V) Faire
    j←j+1
Fin Tant que
Si j>Longeur(V) Alors trouve←vrai Fin Si
Sinon i←i+1
Fin Si
Fin Tant que
# traitement particulier de la dernière chaîne mémorisée dans Tnom
Si NON(trouve) Alors
    Recup_Nom(Tnom,i,TC)
    j←1
    Tant que TC[j]=V[j] ET j<=Longeur(V) Faire
        j←j+1
    Fin Tant que
    Si j>Longeur(V) Alors trouve←vrai Fin Si
Fin Si
Retourner(trouve)
Fin

```

Fin

### 7) 4 pts

Procédure Table\_Symboles(var Tnom:TAB1, var Tpos:TAB2, var Ttype:TAB1)

Variable

i,pos,T:entier

V:chaîne de caractère

Debut

Nb\_Elt(1,NMAX,nb)

Saisie\_Variable(V,T)

pos←1

Ajout\_Variable(V,T,pos,Tnom,Ttype,Tpos)

Pour i de 2 à nb Faire

pos←pos+Longeur(V)+1

répéter

Saisie\_Variable(V,T)

jusqu'à Recherche(V,Tnom,Tpos)=faux

Ajout\_Variable(V,T,pos,Tnom,Ttype,Tpos)

Fin Pour

Tnom[pos+Longeur(V)]←'#'

Fin