



Concours Mathématiques et Physique, Physique et Chimie,
Biologie et Géologie & Technologie
Corrigé de l'épreuve d'Informatique

Barème : EXERCICE 1 : 4 points, EXERCICE 2 : 5 points, PROBLEME : 11 points

Le barème est sur 40

EXERCICE 1 (8 points)

1. $> F := ((x-y-1)*(x^2-y-1))/(2*(x*y-1));$ (0,5)
2. $eval(F, \{x=1, y=3\});$ # {} obligatoires (1,0)
ou bien $subs(\{x=1, y=3\}, F);$ # {} ne sont pas obligatoires
3. $> NF := expand(numer(F));$ # expand obligatoire (1,0)
 $DF := denom(F);$ (0,5)
4. $> nops([coeffs(NF)]);$ # [] obligatoires (0,75)
5. $> quo(NF, DF, x);$ (0,5)
 $rem(NF, DF, x);$ (0,5)
6. $> f := unapply(F, x, y);$ # unapply obligatoire (0,75)
 $f5 := unapply(f(x, 5), x);$ # unapply n'est pas obligatoire (0,5)
7. $> singular(f(x, y));$ (0,5)
8. $> solve(f5(x));$ (0,5)
ou bien $solve(f5(x), x);$
ou encore $solve(f5(x)=0, x);$
9. $> plot(f5, -2*Pi..2*Pi);$ (0,5)
ou bien $plot(f5(x), x=-2*Pi..Pi);$
10. $> plot3d(f, -10..10, -10..10);$ (0,5)
ou bien $plot3d(f(x, y), x=-10..10, y=-10..10);$

EXERCICE 2 (10 points)

1. $evalf(Pi, 20);$ (0,5)
ou bien $Digits:=20; evalf(Pi);$

Remarques :

- L'utilisation de *evalf* dans les procédures Maple suivantes est obligatoire ;
- Les commandes *return* ou *RETURN* ne sont pas obligatoires ;

2. (3,0)

```
> Cues:=proc(epsilon::numeric) (0,5) : entête
local a1,b1,an,bn,df;
a1:=0;b1:=1/4; (0,5) : initialisation
do
  an:=evalf(a1+b1)/2;
  bn:=evalf(sqrt(an*b1));
  df:=abs(1/(2*an)-1/(2*bn)); } (1,0) : traitement
  a1:=an;
  b1:=bn;
  if df <= epsilon then break fi; (0,75) : boucle do ... od avec condition
od;
return(1/(2*an)); (0,25)
end proc;
```

N.B : les valeurs initiales des deux relations de récurrence a_1 et b_1 peuvent être des paramètres en entrée.

3. (2,25)

```
> Brounker:=proc(n::posint) (0,25) : entête
local i,S;
S:=1; (0,25) : initialisation
for i from n to 3 by -2 do (0,75) : boucle for ...
  S:=(2+(i^2)*S)^(-1); (0,75) : traitement
od;
return(evalf(4/(1+S))); (0,25)
end proc;
```

4. $\text{evalf}(6 \cdot \sum_{n=0}^{\infty} \frac{(2^n)!}{(2^{4n+1}) \cdot (n!)^2 \cdot (2^{2n+1})})$; (1,5)

5. (2,75)

```
> Euler:=proc(epsilon::numeric) (0,25) : entête
local k,S,S1;
k:=1;S:=1; (0,5) : initialisation
do
  S1:=evalf(S);
  k:=k+1;
  S:=evalf(S+1/k^2); } (1,25) : traitement
  if abs(S-S1) <= epsilon then break fi; (0,5) : boucle do ... od avec condition
od;
return(evalf(sqrt(6*S))); (0,25)
end proc;
```

PROBLEME (22 points)

Remarques :

- Pour le passage des paramètres E correspond à un passage par valeur et S ou E/S correspondent à un passage par adresse.
- **Retourner (résultat)** correspond à **nom_fonction** \leftarrow *résultat*.

1. (1,0)

FONCTION **SAISIE_NB** (*n*) : entier (0,25) : entête

VARIABLE *n* : entier

DEBUT

REPETER

ECRIRE ("Saisir un entier positif \leq à ", NMAX) } (0,25) : saisie de n

LIRE (*n*)

JUSQU'A (*n* > 0) ET (*n* \leq NMAX) (0,25) : contrôle de la saisie

RETOURNER (*n*) {ou bien SAISIE_NB \leftarrow *n*} (0,25)

FIN

2. (1,5)

PROCEDURE **SAISIE_SEQ** (*E n* : entier, *S T* : TABC) (0,5) : entête

{ou bien PROCEDURE **SAISIE_SEQ** (*n* : entier, VAR *T* : TABC)}

VARIABLE *i* : entier

DEBUT

POUR *i* DE 1 à *n* FAIRE (0,25) : boucle POUR ...

REPETER

ECRIRE ("Saisir un caractère dans l'alphabet A, C, G, T ") } (0,25)

LIRE (*T*[*i*])

JUSQU'A (*T*[*i*] = "A" OU *T*[*i*] = "C" OU *T*[*i*] = "G" OU *T*[*i*] = "T") (0,5) : contrôle

FIN POUR

FIN

3. (1,5)

FONCTION **FCT** (*E n* : entier) : entier (0,5) : entête

VARIABLE *f*, *i* : entier

DEBUT

f \leftarrow 1 (0,25) : initialisation

POUR *i* DE 1 à *n* FAIRE (0,25) : boucle

f \leftarrow *f* * *i* (0,25) : traitement

FIN POUR

RETOURNER (*f*) {ou bien FCT \leftarrow *n*} (0,25)

FIN

4. (1,0)

FONCTION **COMB** (*E n*, *m* : entier) : entier (0,25) : entête

DEBUT

RETOURNER (*FCT*(*n*) / (*FCT*(*m*) * *FCT*(*n* - *m*))) (0,75)

FIN

5. (0,75)PROCEDURE **INIT_TPOS** (E p : entier , S S : TABE) (0,25) : entête{ou bien PROCEDURE **INIT_TPOS** (p : entier , VAR S : TABE)}

VARIABLE i : entier

DEBUT

POUR i DE 1 à p FAIRE (0,25) : boucle

S[i] ← i (0,25) : affectation

FIN POUR

FIN

6. (1,5)PROCEDURE **ECRIRE_TIRETS** (E m , L : entier , E S : TABE , E/S M : MAT) (0,5) : entête{ou bien PROCEDURE **ECRIRE_TIRETS** (m , L : entier , S : TABE , VAR M : MAT)}

VARIABLE i : entier

DEBUT

POUR i DE 1 à p FAIRE (0,25) : boucle

M[L, i] ← "-" (0,75) : affectation

FIN POUR

FIN

7. (2,5)PROCEDURE **ECRIRE_SEQ** (E m , L : entier , E B : TABC , E/S M : MAT) (0,25) : entête

VARIABLE i, j : entier

DEBUT

j ← 1 (0,25) : initialisation {pointer sur la 1^{ère} colonne de M}

POUR i DE 1 à m FAIRE (0,25) : boucle {Parcourir le tableau B}

TANT QUE M[L, j] = "-" FAIRE

j ← j + 1

FIN TANT QUE

M[L, i] ← B[i] (0,75) : affectation

j ← j + 1 (0,25) : incrémentation de j

FIN POUR

FIN

8. (3,75)PROCEDURE **INCREM_SEQ** (E p , n : entier , E/S S : TABE) (0,25) : entête

VARIABLE i, k, x : entier

B : booléen

DEBUT

B ← faux (0,25)

i ← p (0,25)

TANT QUE B = faux ET i >= 1 FAIRE (0,5) : boucle {recherche d'une case}

SI S[i] + (p - i + 1) <= n (0,75) : test sur chaque case

ALORS x ← S[i] (0,25)

POUR k DE 1 à (p - i + 1) FAIRE (0,5)

S[i + k - 1] ← x + k (0,5)

FIN POUR

B ← vrai (0,25)

SINON i ← i - 1 (0,25)

FIN SI

FIN TANT QUE

FIN

9. (3,5)

PROCEDURE **CREE_MAT** ($\underline{E} n, m : \text{entier}, \underline{E} B : \text{TABC}, \underline{S} q : \text{entier}, \underline{S} M : \text{MAT}$) (0,5)

VARIABLE $i, j, p : \text{entier}$
 $S : \text{TABE}$ } (0,25) : déclaration de variables locales

DEBUT

$p \leftarrow n - m$ (0,25) {n'est pas obligatoire}

$q \leftarrow \text{COMB}(n, p)$ (0,25) : nombre de lignes de M

POUR i DE 1 à q FAIRE
 POUR j DE 1 à n FAIRE
 $M[i, j] \leftarrow "x"$
 FIN POUR
 FIN "POUR" } (0,5) : initialisation de M

INIT_TPOS (p, S) (0,25)

ECRIRE_TIRETS (p, l, S, M) (0,25)

ECRIRE_SEQ (m, l, B, M) (0,25)

POUR i DE 2 à q FAIRE (0,25)

INCREM_SEQ (p, n, S) (0,25)

ECRIRE_TIRETS (p, i, S, M) (0,25)

ECRIRE_SEQ (m, i, B, M) (0,25)

FIN POUR

FIN

10. (2,25)

FONCTION **SCORE** ($\underline{E} n, L : \text{entier}, \underline{E} A : \text{TABC}, \underline{E} M : \text{MAT}$) : entier (0,25)

VARIABLE $i, Sc : \text{entier}$

DEBUT

$Sc \leftarrow 0$ (0,25)

POUR i DE 1 à n FAIRE (0,25)
 SI $M[L, i] = "-"$ ALORS $Sc \leftarrow Sc - 2$
 SINON SI $M[L, i] = A[i]$ ALORS $Sc \leftarrow Sc + 2$
 SINON $Sc \leftarrow Sc - 1$ } (1,25)

FIN SI

FIN SI

FIN POUR

RETOURNER (Sc) (0,25)

FIN

11. (2,75)

FONCTION **SCORE_OPT** ($\underline{E} n, q : \text{entier}, \underline{E} A : \text{TABC}, \underline{E} M : \text{MAT}$) : entier (0,25)

VARIABLE $MSc, i : \text{entier}$

DEBUT

$MSc \leftarrow \text{SCORE}(n, 1, A, M)$ (0,75) : calcul de score de la 1^{ère} ligne

POUR i DE 2 à q FAIRE (0,25) : boucle {parcourir les lignes restantes}

SI $MSc < \text{SCORE}(n, i, A, M)$
 ALORS $MSc \leftarrow \text{SCORE}(n, i, A, M)$ } (1,25) : traitement

FIN SI

FIN POUR

RETOURNER (MSc) (0,25)

FIN