

# Correction Algorithmique et programmation

## Session de contrôle 2009

### Exercice 1 :

**NB: On acceptera toute proposition équivalente.**

1. La trace d'exécution de la fonction f pour les cas suivants :

➤ T : 

5	7	6	3
---	---	---	---

 ; m = 6 et n = 4

f(6,4,T) = f(6,3,T) = vrai

➤ T : 

5	7	6	3
---	---	---	---

 ; m = 1 et n = 4

f(1,4,T) = f(1,3,T) = f(1,2,T) = f(1,1,T) = f(1,0,T) = faux

2. La fonction f permet de vérifier l'existence d'un entier m dans un tableau T de n entiers

3. Forme récursive de la fonction f (1 pt)

0) DEF FN f(m,n : entier ; T : vect) : booléen

1) Si n = 0 Alors

f ← faux

Sinon Si T[n]=m alors f ← vrai

Sinon f ← f(m,n-1,T)

FinSi

2) Fin f

0,25

0,25

0,25

0,25

4. Forme itérative de la fonction f (1 pt)

0) DEF FN f(m,n : entier ; T : vect) : booléen

1) f ← faux ; i ← n+1

Répéter

[ i ← i - 1] Si T[[i] = m Alors

f ← vrai

FinSi

Jusqu'à (f = vrai) ou (i=1)

2) Fin f

0,25

0,25

0,25

0,25

### Exercice 2 :

1) Analyse du module de calcul de la valeur de X: ( 2 pts)

DEF FN Piece: réel	0,5
Résultat = Piece ← X	0,5
X=[X ← 0, S2 ← 0 ]	0,25
Répéter	
X ← X+0,01	0,25
S1 ← S2	
S2 ← ((√3 - 4)/2)*X <sup>2</sup> +X	0,25
Jusqu'à  S2-S1  ≤ 0	0,25

NB: Il est possible de trouver une autre solution basée sur le calcul de la dérivée de S(x).

L'aire S de la pièce est égale à  $S = ((\sqrt{3} - 4) / 2) * x^2 + x$ .

L'aire S est maximale ⇒ la dérivée s'annule ⇒  $2 * ((\sqrt{3} - 4) / 2) * x + 1 = 0$

⇒  $x = -1 / (2 * ((\sqrt{3} - 4) / 2))$

**Tableau de déclaration des objets globaux**

Objet	Type/Nature	Rôle
x	réel	valeur du côté du triangle

**Algorithme :**

- 0) DEF FN Piece :Réal
- 1)  $X \leftarrow 0, S2 \leftarrow 0$
- 2) Répéter
  - $X \leftarrow X + 0,01$
  - $S1 \leftarrow S2$
  - $S2 \leftarrow ((\sqrt{3} - 4)/2) * X^2 + X$
  - Jusqu'à  $S2 - S1 \leq 0$
- 3) Piece  $\leftarrow X$
- 4) Fin Piece

2

**Problème :****Analyse du programme principal**

Nom : calcul\_integral

Résultat = écrire ("VII par la méthode des trapèzes : ", VI1,  
 "VI2 par la méthode d'une subdivision aléatoire : ", VI2,  
 "La valeur absolue de la différence est ", | VI1 - VI2 | )

VI1  $\leftarrow$  FN TRAPEZES(n)VI2  $\leftarrow$  FN SUBDIVISIONS(n)

Répéter

N= donnée("introduire le nombre de subdivisions : ")

Jusqu'à n dans [101..999]

Fin calcul\_integral

**Tableau de déclaration des objets globaux**

Objet	Type/Nature	Rôle
VI1	Réal	Valeur de l'intégral par la méthode des trapèzes
VI2	Réal	Valeur de l'intégral par méthode d'une subdivision aléatoire
N	Entier	Nombre de subdivisions
TRAPEZES	Fonction	Fonction qui calcule selon la méthode des trapèzes, une valeur approchée de l'aire en question
SUBDIVISIONS	Fonction	Fonction qui calcule selon la méthode des subdivisions aléatoires, une valeur approchée de l'aire en question

**Algorithme du programme principal**

- 0) Début calcul\_integral
- 1) Répéter
  - Ecrire("introduire le nombre de subdivisions : ")
  - Lire(n)
  - Jusqu'à n dans [101..999]
- 2) VI1  $\leftarrow$  FN TRAPEZES(n)
- 3) VI2  $\leftarrow$  FN SUBDIVISIONS(n)
- 4) écrire ("VII par la méthode des trapèzes : ", VI1, "VI2 par méthode d'une subdivision aléatoire : ", VI2, "La valeur absolue de la différence est ", | VI1 - VI2 | )
- 5) Fin calcul\_integral

## Analyse de la fonction TRAPEZES

DEF FN TRAPEZES (n : entier) : réel

Résultat = TRAPEZES ← VI

VI ← ((exp(-1) + exp(-4))/2)/n Pour i de 1 à n-1 faire

VI ← VI + (exp(-(1+i/n)<sup>2</sup>))/n

FinPour

i = compteur

Fin TRAPEZES

**Tableau de déclaration des objets de la fonction TRAPEZES**

Objet	Type/Nature	Rôle
VI	Réel	Valeur de l'intégral par la méthode des trapèzes
I	Entier	compteur

### Algorithme de la fonction TRAPEZES

0) DEF FN TRAPEZES (n : entier) : réel

1) VI ← ((exp(-1) + exp(-4))/2)/n

Pour i de 1 à n-1 faire

VI ← VI + (exp(-(1+i/n)<sup>2</sup>))/n

FinPour

2) TRAPEZES ← VI

3) Fin TRAPEZES

### Analyse de la fonction SUBDIVISIONS

	Commentaires
DEF FN SUBDIVISIONS (n : entier) : réel	
Résultat = SUBDIVISIONS ← VI	
VI ← (S1+S2)/2	• Détermination de la valeur de I2
S1=[S1 ← 0] Pour i de 0 à n-1 faire	
S1 ← S1 + (V[i+1] - V[i]).exp(-(V[i]) <sup>2</sup> )	• Détermination de la somme S1
FinPour	
S2=[S2 ← 0] Pour i de 0 à n-1 faire	
S2 ← S2 + (V[i+1] - V[i]).exp(-(V[i+1]) <sup>2</sup> )	• Détermination de la somme S2
FinPour	
i = compteur	
V = PROC REMPLISSAGE (n, V)	• V est un tableau rempli aléatoirement
PROC TRI (n, V)	puis trié par la méthode d'insertion.
Fin SUBDIVISIONS	

**Tableau de déclaration des nouveaux types**

Type
Vect = tableau de 999 réels

**Tableau de déclaration des objets de la fonction SUBDIVISIONS**

Objet	Type/Nature	Rôle
VI	Réel	Valeur de l'intégral par la méthode des subdivisions
S1	Réel	Valeur de la somme S1
S2	Réel	Valeur de la somme S2
i	Entier	Compteur
V	Vect	Tableau contenant les valeurs des x <sub>i</sub>
Tri	Procédure	Une procédure qui permet de trier les éléments de V par la méthode d'insertion

**Algorithme de la fonction SUBDIVISIONS**

- 0) DEF FN SUBDIVISIONS (n:entier) : réel
- 1) PROC REEMPLISSAGE(n,V)
- 2) PROC TRI(n,V)
- 3) S1 ← 0
  - Pour i de 0 à n-1 faire
    - S1 ← S1 + (V[i+1] - V[i]).exp(-(V[i])<sup>2</sup>)
  - FinPour
- 4) S2 ← 0
  - Pour i de 0 à n-1 faire
    - S2 ← S2 + (V[i+1] - V[i]).exp(-(V[i+1])<sup>2</sup>)
  - FinPour
- 5) VI ← (S1+S2)/2
- 6) SUBDIVISIONS ← VI
- 7) Fin SUBDIVISIONS

**Analyse de la procédure TRI**

DEF PROC TRI (n : entier ; Var T :Vect)	Commentaires
Résultat = T	
T = [] Pour i de 1 à n faire	
Aux ← T[i]	• Application du tri par insertion sur un tableau T
j ← i	
Tant que (T[j-1] > aux) et (j > 1) Faire	
T[j] ← T[j-1]	
j ← j-1	
Fin Tant que	
T[j] ← aux	
FinPour	
i, j = compteurs	
Fin TRI	

**Tableau de déclaration des objets de la Procédure TRI**

Objet	Type/Nature	Rôle
i	Entier	Compteur
j	Entier	Compteur
aux	Réel	Variable auxiliaire

**Algorithme de la procédure TRI**

- 0) DEF PROC TRI (n : entier ; Var T :Vect)
- 1) Pour i de 1 à n faire
  - Aux ← T[i]
  - j ← i
  - Tant que (T[j-1] > aux) et (j > 1) Faire
    - T[j] ← T[j-1]
    - j ← j-1
  - Fin Tant que
  - T[j] ← aux
- FinPour
- 2) Fin TRI

## Analyse de la procédure REMPLISSAGE

DEF PROC REMPLISSAGE (n : entier ; Var T :Vect)

Résultat = T
T=[T[0] ← 1 ; T[n] ← 2]
Pour i de 1 à n-1 faire
Répéter
T[i] ← 1 + RANDOM
Jusqu'à Vérif (i,T)
FinPour
i = compteur
Fin REMPLISSAGE

### Commentaires

- Cette procédure permet de générer la suite  $(X_i)_{0 \leq i < n}$  où  $X_0 = 1$ ,  $X_i = V[i]$  et  $X_n = 2$ .
- Puisque la fonction RANDOM renvoie un réel compris entre **0 et 1 au sens strict**, il faut prévoir de ranger dans **T**, les **2** valeurs (**1 et 2**) pour construire la suite  $X_i$
- La valeur obtenue au hasard (**1+ random**), ne sera considérée que si elle **n'existe pas dans le tableau**. (car selon l'énoncé, V contient n-1 réels distincts)

Tableau de déclaration des objets de la Procédure REMPLISSAGE

Objet	Type/Nature	Rôle
i	Entier	Compteur
X	Réel	L'entier à déterminer au hasard
Vérif	Fonction	Fonction qui vérifie l'existence d'un élément dans les I premiers éléments de T

### Algorithme de la procédure REMPLISSAGE

```

0) DEF PROC REMPLISSAGE (n : entier ; Var T :Vect)
1) T[0] ← 1
   T[n] ← 2
   Pour i de 1 à n-1 faire
       Répéter
           T[i] ← 1 + RANDOM
       Jusqu'à Vérif (i,T)
   FinPour
2) Fin REMPLISSAGE
    
```

### Analyse de la fonction Vérif

DEF FN VERIF ( j : entier ; T :Vect) : booléen
Résultat = Vérif ← V
V= [V ← Vrai, x ← T[j] ] Répéter
Si x = T[j-1] alors V ← faux
Sinon j ← j-1
FinSi
Jusqu'à (V=faux) ou (j=1)
Fin VERIF

### Commentaires

- On aurait pu exploiter la fonction f de l'exercice 1
- Cette fonction vérifie que la valeur T[j] n'existe pas dans le tableau.

### Algorithme de la fonction Vérif

```

0) DEF FN VERIF ( j : entier ; T :Vect) : booléen
1) V ← Vrai, x ← T[j]
   Répéter
       Si x = T[j-1] alors V ← faux
       Sinon j ← j-1
   FinSi
   Jusqu'à (V=faux) ou (j=1)
2) Vérif ← V
3) Fin VERIF
    
```

## Barème du problème

	<b>ANALYSES ( 8 pts)</b>	<b>ALGO (4 pts) / -0,25 par erreur</b>
PP	1,5 pts = 0,5 cohérence + 1 modularité	0,75 pt
Affichage de $I_1$ , $I_2$ et $ I_1-I_2 $	0,75 pts = 3 x 0,25	0,5 pt
Calcul de $I_1$	1 pt = 0,25 init + 0,25 boucle Pour + 0,25 $V_i \leftarrow V_i \dots$ + 0,25 Trapèzes $\leftarrow V_i$	0,5 pt
Calcul de $I_2$	1,25 pt = 0,5 $S_1$ + 0,5 $S_2$ + 0,25 appels tri et remp	0,5 pt
Tri de V	1 pt = 0,25 boucle Pour + 0,25 init + 0,25 décalage + 0,25 Insertion	0,5 pt
Remplissage de V	1pt = 0,5 remplissage + 0,5 Vérification	1 pt = 0,5 + 0,5
Saisie de N	0,5pt = 0,25 lecture + 0,25 répéter et condition	0,25 pt
Divers :		
- TDO	0,5 pt	
- Paramètres et modes de passages	0,5 pt	