Section : .... N° d'inscription : .... Série :	Signature des surveillants
Nom et Prénom :	....
Date et lieu de naissance :	....

✂ --

Algorithmique et Programmation - Section : Sciences de l'informatique – Session de contrôle 2025

Le sujet comporte 5 pages numérotées de 1/5 à 5/5.

Les réponses aux questions de l'exercice 1 doivent être rédigées sur les pages 1/5 et 2/5 qui doivent être remises à la fin de l'épreuve.

Exercice 1 (3 points)

……./3,00

Soit l'algorithme suivant de la fonction **Inconnue** :

```
Fonction Inconnue (N, B : Entier) : Chaine de caractères
DEBUT
    Si N=0 Alors
            Retourner "0"
    Sinon
        CH ← ""
        Répéter
            K ← N Mod B + 48 + ((N Mod B) Div 10) * 7
            CH ← Chr (K) + CH
            N ← N Div B
        Jusqu'à N = 0
        Retourner CH
    FinSi
FIN
```

NB : Chr(65) retourne le caractère **"A"**, **Chr(97)** retourne le caractère **"a"** et **Chr(48)** retourne le caractère **"0"**.

Travail demandé

1) Valider chacune des propositions suivantes en mettant dans la case correspondante la lettre « **V** » si elle est correcte ou la lettre « **F** » dans le cas contraire.

a) Le résultat retourné par l'appel **Inconnue(5, 2)** est :

☐ 100 ………..

☐ 31 ………..

☐ 101 ………..

b) Le résultat retourné par l'appel **Inconnue(31, 16)** est :

☐ 1F ………..

☐ 31 ………..

☐ F1 ………..

Ne rien écrire ici

c) La fonction **Inconnue** permet de :

☐ convertir un nombre hexadécimal en décimal. …………

☐ convertir un nombre décimal vers une base donnée. …………

☐ convertir un nombre d'une base donnée vers la base 10. …………

2) Proposer un algorithme récursif pour la fonction **Inconnue**. …………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

Exercice 2 (3,5 points)

Soit **K** un entier de **d** chiffres (avec d ≥ 2) ayant la forme **$C_1C_2 \ldots C_d$**. A partir de ce nombre on définit une suite **U** telle que les **d** premiers termes sont les **d** chiffres du nombre **K**. Ensuite chaque terme suivant est calculé en additionnant les **d** termes précédents.

$$U\begin{cases} U_1 = C_1 \\ U_2 = C_2 \\ \ldots\ldots \\ U_d = C_d \\ U_n = U_{n-1} + U_{n-2} + \cdots + U_{n-d} \quad pour\ n > d \end{cases}$$

Le nombre **K** est appelé **nombre de Keith**, si à un moment donné, un terme de la suite **U** est égal au nombre **K**.

Exemples :

- Pour **K = 197**, le nombre de chiffres qui le forment est égal à **3**, donc **d = 3** et la suite **U** est définie par :

$$U\begin{cases} U_1 = 1 \longleftarrow \text{Le 1}^{\text{er}}\text{ chiffre de K} \\ U_2 = 9 \longleftarrow \text{Le 2}^{\text{ème}}\text{ chiffre de K} \\ U_3 = 7 \longleftarrow \text{Le 3}^{\text{ème}}\text{ chiffre de K} \\ U_n = U_{n-1} + U_{n-2} + U_{n-3} \quad pour\ n > 3 \end{cases}$$

  Les termes de la suite sont : U_1=1, U_2=9, U_3=7, U_4=17, U_5=33, U_6=57, U_7=107, U_8=<u>197</u>, ...
  U_8 = 197 = K, **K** est un terme de la suite donc **K est un nombre de Keith**.

- Pour **k = 24**, le nombre de chiffres qui le forment est égal à **2**, donc **d = 2** et la suite **U** est définie par :

$$U\begin{cases} U_1 = 2 \longleftarrow \text{Le 1}^{\text{er}}\text{ chiffre de K} \\ U_2 = 4 \longleftarrow \text{Le 2}^{\text{ème}}\text{ chiffre de K} \\ U_n = U_{n-1} + U_{n-2} \quad pour\ n > 2 \end{cases}$$

  Les termes de la suite sont : U_1=2, U_2=4, U_3=6, U_4=10, U_5=16, U_6=26, ... Comme **U_6=26>K** donc **K** n'est pas un terme de la suite et par conséquent **K n'est pas un nombre de Keith**.

Travail demandé :

1) Ecrire un algorithme d'une fonction **Verif(K)** qui permet de retourner :
- La valeur de **n** telle que **U_n=K**, si **K** est un **nombre de Keith**
- La valeur **-1** dans le cas contraire.

Exemples : Verif(197) retourne **8** et **Verif(24)** retourne **-1**

2) En utilisant la fonction **Verif**, écrire un algorithme d'un programme qui permet de remplir un tableau d'enregistrements **NK** par les nombres de **Keith** qui sont inférieurs à **100000**, sachant que chaque enregistrement du tableau **NK** contient les champs suivants :
- **Nbre** : le nombre de **Keith**.
- **Num** : le numéro du terme de la suite **U** qui est égal au nombre de **Keith** contenu dans le champ **Nbre**.

Exercice 3 (3,5 points)

Une matrice carrée **M1** de taille **d*d** est dite **Matrice multiple d'un premier** si le plus grand commun diviseur (**PGCD**) de ses éléments est un nombre premier.

Pour calculer le **PGCD** des éléments de la matrice **M1**, on suit les étapes suivantes :

- **Etape 1** : Stocker tous les éléments de la matrice **M1** dans la première ligne d'une matrice **M2** de taille $\mathbf{d^2 * d^2}$.
- **Etape 2** : Remplir les autres lignes de la matrice **M2** comme suit :
  M2[L,C] = PGCD(M2[L-1,C], M2[L-1,C+1]) .

Dans ce cas, le **PGCD** des éléments de **M1** est le contenu de la case **M2[$\mathbf{d^2}$ - 1,0]**

Exemple : Pour **d = 2** et la matrice **M1** suivante :

M1

	0	1
0	20	10
1	4	8

On génère la matrice **M2** suivante :

M2

	0	1	2	3
0	20	10	4	8
1	10	2	4	
2	2	2		
3	2			

En effet : - On remplit la ligne d'indice **0** par les éléments de M1.
- On remplit les autres lignes comme suit :

La ligne d'indice 1 :
M2[1,0] = PGCD(20,10) = **10**
M2[1,1] = PGCD(10,4) = **2**
M2[1,2] = PGCD(4,8) = **4**

La ligne d'indice 2 :
M2[2,0] = PGCD(10,2) = **2**
M2[2,1] = PGCD(2,4) = **2**

La ligne d'indice 3 :
M2[3,0] = PGCD(2,2) = **2**

Le **PGCD** des éléments de **M1** est égal à **M2[3,0]** qui est égal à **2**. Comme **2** est un nombre premier, donc **M1** est dite **Matrice multiple d'un premier**.

Travail demandé

1) Ecrire un algorithme d'une fonction **PGCD(a, b)** qui permet de calculer le **PGCD** de deux entiers **a** et **b**.

2) En appliquant les étapes définies précédemment et en utilisant la fonction **PGCD** de la question **1)**, écrire un algorithme d'une fonction **Verif(M1, d)** qui permet de vérifier si la matrice **M1** de taille **d*d** est dite **Matrice multiple d'un premier**.

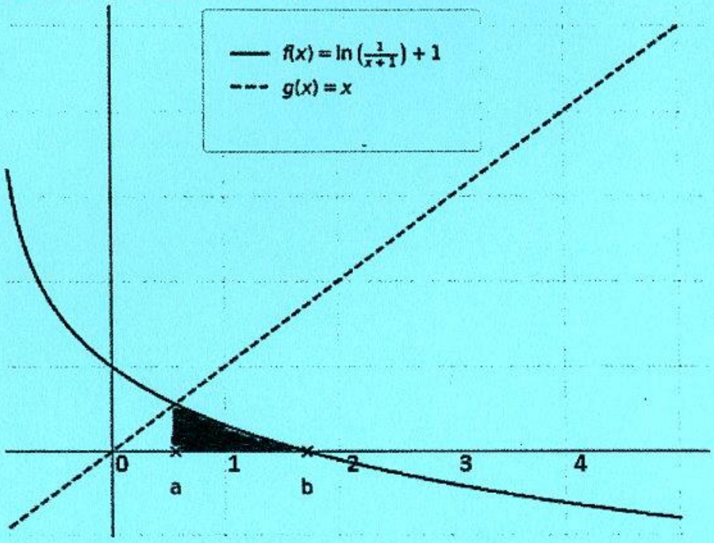
   NB :
   - **M1** est de type **Mat1**.
   - **M2** est de type **Mat2**.
   - Le candidat n'est pas appelé à dresser le tableau de déclaration pour définir les types **Mat1** et **Mat2**.

Exercice 4 (10 points)

Soient les deux fonctions f et g suivantes :

- $f(x) = Ln\left(\frac{1}{x+1}\right) + 1$, définie et continue sur l'intervalle $]-1,+\infty[$ (Ln désigne le logarithme népérien).
- $g(x) = x$, définie et continue sur $\mathbb{R}$.

Ci-après les représentations graphiques des deux fonctions :



Avec a et b sont :

- a : Le point fixe de la fonction f, f(a) = a.
- b : Le point zéro de la fonction f, f(b) = 0 *dans l'intervalle [1, 2].*

Soit A = $\int_a^b f(x).dx$ l'aire de la partie grisée. Afin d'étudier la convergence des approximations de cette aire par les méthodes "**rectangles à gauche**" et "**rectangles à droite**" vers celle des "**rectangles du point milieu**", on se propose d'écrire un programme permettant de réaliser les tâches suivantes :

- La saisie d'un réel **epsilon** positif et inférieur ou égal à $\mathbf{10^{-4}}$.
- La recherche du point fixe de f (le point a) à **epsilon** près.
- La recherche du point zéro de f (le point b) à **epsilon** près.
- La génération d'un fichier d'enregistrements "**Calcul.dat**" sur la racine du disque **C** où chaque enregistrement est formé par les champs suivants :
  - **n** : le nombre des subdivisions dans l'intervalle [a, b].
  - **rd** : l'aire A calculée par la méthode des rectangles à droite avec **n** subdivisions.
  - **rg** : l'aire A calculée par la méthode des rectangles à gauche avec **n** subdivisions.
  - **rm** : l'aire A calculée par la méthode des rectangles du point milieu avec **n** subdivisions.

  NB :
  - Le traitement s'arrête lorsque la moyenne des aires calculées par les deux méthodes rectangles à droite et rectangles à gauche converge vers l'aire calculée par la méthode des rectangles du point milieu à epsilon près, c'est à dire :

    $\left|\frac{rd+rg}{2} - rm\right| < epsilon$
  - **n** est initialisé à **1** et s'incrémente jusqu'à la condition d'arrêt.
  - Le candidat peut utiliser la fonction **Ln(*x*)**, sans la développer, pour calculer le logarithme népérien de ***x***.

Travail demandé :

1) Ecrire un algorithme du programme principal en le décomposant en modules permettant de réaliser les tâches décrites précédemment.

2) Ecrire un algorithme pour chaque module envisagé.

RÉPUBLIQUE TUNISIENNE MINISTÈRE DE L'ÉDUCATION EXAMEN DU BACCALAURÉAT SESSION 2025	**Session de contrôle 2025**	
	Épreuve : ***Algorithmique et Programmation***	Sections : ***Sciences de l'informatique***
	Durée : **3h**	Coefficient de l'épreuve: **2**

Corrigé et barème de notation

Barème de notation sur 40 points, arrondir la note au 2ème chiffre après la virgule.

N.B :

- **On accepte toute autre solution algorithmique correcte qui respecte les recommandions officielles**
- **Retrancher 0.25 par erreur**

Exercice 1 (6 points = 4.5 + 1.5)

1) 4.5 points = 0.5*9

a) Le résultat retourné par l'appel **Inconnue(5,2)** est :

F	100
F	31
V	101

b) Le résultat retourné par l'appel **Inconnue(31,16)** est :

V	1F
F	31
F	F1

c) La fonction **Inconnue** permet de :

F	convertir un nombre Hexadécimal en décimal.
V	convertir un nombre décimal vers une base donnée.
F	convertir un nombre d'une base donnée vers la base 10.

2) 1.5 points

N.B. : - Accepter uniquement une solution récursive.
- Retrancher 0.25 par erreur y compris le TDOL

```
Fonction Inconnue (N, B : Entier) : Chaine de caractères
DEBUT
   Si N=0 Alors
                  Retourner "0"
   Sinon
                  K ← N Mod B + 48 + ((N Mod B) Div 10) * 7
                  Si (N Div B = 0) Alors
                         Retourner Chr(K)
                  Sinon
                         Retourner  Inconnue(N Div B, B) + Chr (K)
                  FinSi
   FinSi
FIN
```

TDOL

Objet	Type/Nature
K	Entier

Exercice 2 : (7 points = 4.25 + 2.75)

1) *4.25 points*

```
Fonction Verif (k : Entier) : Entier                0.25
DEBUT
    ch ← Convch (k)                                 0.25
    n ← Long(ch)                                    0.25
    Pour i de 0 à n-1 Faire                         0.25
            T[i] ← Valeur(ch[i])                    0.25
    Fin Pour
    Répéter
            s ← 0                                   0.25
            Pour j de i-n+1 à i Faire               0.5
                    s ← s + T[j]                    0.25
            Fin Pour
            i ← i+1                                 0.25
            T[i] ← s                                0.25
    Jusqu'à s ≥ k                                   0.25
    Si s=k Alors                                    0.25
            Retourner i+1                           0.25
    Sinon
            Retourner -1                            0.25
    FinSi
FIN
```

TDOL (**0.5**)

Objet	Type/Nature
ch	Chaine de caractères
T	Tableau de 100 entiers
n	Entier
i, j	Entier
s	Entier

2) *2.75 points*

```
Algorithme Keith
DEBUT
        Nb ← 0                                      0.25
        Pour k de 10 à 100000 Faire                 0.25
            nm ← Verif(k)                           0.25
            Si nm ≠ -1 Alors                        0.25
                        e.Nbre ← k                  0.25
                        e.Num ← nm                  0.25
                        NK[Nb] ← e                  0.25
                        Nb ← Nb + 1                 0.25
            FinSi
        Fin Pour
FIN
```

TDNT(**0.5=0.25*2**)

Type
Enreg = Enregistrement Nbre, Num : Entier Fin
Tab = Tableau de 100000 Enreg

TDO (**0.25**)

Objet	Type/Nature
k, Nb, nm	Entier
e	Enreg
NK	Tab
Verif	Fonction

Exercice 3 (7 points)

1) 2 points = (Entête : 0.5 + traitement : 1.25 + retour du résultat : 0.25)

```
Fonction PGCD(a,b : Entier) : Entier
DEBUT
    Tant que (a ≠ b) Faire
        Si a>b Alors
            a ← a-b
        Sinon
            b ← b-a
        FinSi
    Fin Tant que
    Retourner a
FIN
```


2) 5 points (0.5 : pour tous les TDOLs)

```
Fonction Verif(M1 : Mat1 , d : Entier) : Booléen                0.5
DEBUT
    Repartir (M1, M2, d)
    Pour i de 1 à d*d-1 Faire                                   0.25
        Pour j de 0 à d*d-i-1 Faire                             0.25
            M2[i,j] ← PGCD(M2[i-1,j], M2[i-1,j+1])              0.5
        Fin Pour
    Fin Pour
    Retourner Premier(M2[d*d-1,0])                              0.5
FIN
```

TDOL

Objet	Type/Nature
i	Entier
j	Entier
M2	Mat2
Repartir	Procédure
PGCD	Fonction
Premier	Fonction

```
Procédure Repartir (M1 : Mat1, @ M2 : Mat2, d : Entier)
DEBUT
    Pour i de 0 à d-1 Faire                                     0.25
        Pour j de 0 à d-1 Faire                                 0.25
            M2[0,i*d+j] ← M1[i,j]                               0.75
        Fin Pour
    Fin Pour
FIN
```

TDOL

Objet	Type/Nature
i, j	Entier

```
Fonction Premier (x : Entier) : Booléen
DEBUT
    i ← 1
    Répéter
        i ← i +1
    Jusqu'à (x Mod i= 0) Ou (i ≥ Racinecarré(x))                1.25
    Retourner (i > Racinecarré(x) et (x≠1))
FIN
```

TDOL

Objet	Type/Nature
i	Entier

Exercice 4 : (20 points)

```
Algorithme Calcul_aire
DEBUT
        Saisie(epsilon)
        a ← Point_fixe (epsilon)
        b ← Point_zero (epsilon)
        Generer (a, b, epsilon)
FIN
```

TDOG

Objet	Type / Nature
a, b	Réel
epsilon	Réel
Saisie	Procédure
Point_fixe	Fonction
Point_zero	Fonction
Generer	Procédure

```
Procédure Saisie (@ epsilon : Réel)
DEBUT
        Répéter
                Ecrire("Donner epsilon")
                Lire (epsilon)
        Jusqu'à (0< epsilon ≤ 0.0001)
FIN
```

2025 جويلية 02

```
Fonction Point_fixe (epsilon : Réel) : Réel
DEBUT
        xa ← 0
        Répéter
                xp ← xa
                xa ← f (xp)
        Jusqu'à (Abs (xa-xp) ≤ epsilon)
        Retourner xa
FIN
```

TDOL

Objet	Type / Natue
xp, xa	Réel
f	Fonction

```
Fonction f (x : réel) : réel
Début
        Retourner Ln(1/(1+x))+1
FIN
```

TDOL

Objet	Type / Natue
Ln	Fonction

```
Fonction point_zero (epsilon : Réel) : Réel
DEBUT
         a ←1
         b←2
         Répéter
                m← (a+b)/2
                Si (f(a)*f(m) > 0 alors
                        a ←m
                Sinon
                        b ←m
                FinSi
        Jusqu'à  (f(m) = 0) Ou (b-a < epsilon)
        Retourner m
FIN
```

TDOL

Objet	Type / Natue
m, a, b	Réel
f	fonction

```
Procédure Generer (a, b, epsilon : Réel)
DEBUT
        Ouvrir ("Calcul.dat", Fc, "wb")
        e.n←0
        Répéter
            e.n ←e.n+1
            e.rd ← rectangleD (a, b, e.n)
            e.rg ← rectangleG(a, b, e.n)
            e.rm ← rectangleM(a, b, e.n)
            ecrire(Fc, e)
        Jusqu'à Abs(((e.rd+e.rg)/2)-e.rm) ≤ epsilon
        Fermer(F)
FIN
```

TDOL

Objet	Type / Nature
e	Enreg
Fc	Fich

TDNT

Type
Enreg= Enregistrement N, rd, rg, rm : Réel FIN
Fich = Fichier des Enreg

```
Fonction rectangleD (a, b : Réel , n : Entier) : Réel
DEBUT
        s ← 0
        h ←(b-a)/n
        x ← a+h
        Pour i de 1 à n faire
            s ←s+f(x)
            x ←x+h
        Fin Pour
        Retourner s*h
FIN
```

TDOL

Objet	Type / Nature
s, h, x	Réel
f	Fonction

02 جويلية 2025

```
Fonction rectangleG (a, b : Réel , n : Entier) : Réel
DEBUT
        s ← 0
        h ←(b-a)/n
        x ← a
        Pour i de 1 à n faire
            s ←s+f(x)
            x ←x+h
        Fin Pour
        Retourner s*h
FIN
```

TDOL

Objet	Type / Nature
s, h, x	Réel
f	Fonction

```
Fonction rectangleM (a, b : Réel , n : Entier) : Réel
Début
        s ← 0
        h ←(b-a)/n
        x ← a+h/2
        Pour i de 1 à n faire
            s ←s+f(x)
            x ←x+h
        Fin Pour
        Retourner s*h
Fin
```

TDOL

Objet	Type / Nature
s, h, x	Réel
f	Fonction

Barème : 20 points

Modularité	***2***
Cohérence	***2***
TDNT + TDOG	***1 =*** *0.5 + 0.5*
TDOL	***1***
Saisie de **epsilon** *(Lecture + Boucle)*	***1.75 =*** *0.75 + 1*
Détermination du point fixe • *Initialisation* • *Boucle* • *Affectations* • *Retour du résultat*	***3=*** *0.5* *1* *1= 0.5*2* *0.5*
Détermination du point zéro • *Initialisations* • *Boucle* • *Affectation* • *Instruction Si* • *Retour du résultat*	***3.25 =*** *0.5=0.25*2* *1* *0.5* *0.75* *0.5*
Génération du fichier • *Calcul d'aire (3 méthodes)* • *Boucle* • *Détermination des champs de l'enregistrement* • *Ecriture dans le fichier* • *Ouverture et fermeture du fichier*	***6 =*** *3 = 1*3* *1* *1 =0.25*4* *0.5* *0.5=0.25*2*

N.B. *: Retrancher 0.25 par erreur*

2025 جويلية 02