Question 4 :

Soit deux réels a et b avec $a < b$.
On considère une fonction f définie, continue, strictement croissante sur l'intervalle $[a\,;\,b]$ et qui s'annule en un réel α.
Parmi les propositions suivantes, la fonction en langage Python qui permet de donner une valeur approchée de α à 0,001 est :

a.
```
def racine(a, b) :
    while abs(b - a) >= 0.001 :
        m = (a + b)/2
        if f(m) < 0 :
            b = m
        else :
            a = m
    return m
```

b.
```
def racine(a, b) :
    m = (a + b)/2
    while abs(b - a) >= 0.001 :
        if f(m) < 0 :
            a = m
        else :
            b = m
    return m
```

c.
```
def racine(a, b) :
    m = (a + b)/2
    while abs(b - a) <= 0.001 :
        if f(m) < 0 :
            a = m
        else :
            b = m
    return m
```

d.
```
def racine (a, b) :
    while abs (b - a) >= 0.001 :
        m = (a + b)/2
        if f(m) < 0 :
            a = m
        else :
            b = m
    return m
```

---

5. On considère la fonction `mystere` définie ci-dessous qui prend une liste L de nombres en paramètre.

On rappelle que `len`(L) représente la longueur de la liste L.

```
def mystere(L) :
    S = 0
    for i in range(len(L)) :
        S = S + L[i]
    return S / len(L)
```

Affirmation : L'exécution de `mystere`([1, 9, 9, 5, 0, 3, 6, 12, 0, 5]) renvoie 50.

Question 5 :

On pose $S = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \ldots + \frac{1}{100}$.
Parmi les scripts Python ci-dessous, celui qui permet de calculer la somme S est :

a.
```
def somme_a() :
    S = 0
    for k in range(100) :
        S=1/(k+1)
    return S
```

b.
```
def somme_b() :
    S = 0
    for k in range(100) :
        S = S + 1/(k + 1)
    return S
```

c.
```
def somme_c() :
    k = 0
    while S < 100 :
        S = S+1/(k+1)
    return S
```

d.
```
def somme_d() :
    k = 0
    while k < 100 :
        S = S + 1/(k + 1)
    return S
```

---

5. On considère la fonction **mystere** définie ci-dessous qui prend une liste L de nombres en paramètre.

On rappelle que `len(L)` renvoie la longueur, c'est-à-dire le nombre d'éléments de la liste L.

```
def mystere(L) :
    M = L[0]
    # On initialise M avec le premier élément de la liste L
    for i in range(len(L)) :
        if L[i] > M :
            M = L[i]
    return M
```

Affirmation : L'exécution de **mystere**([2, 3, 7,0 ,6, 3, 2, 0, 5]) renvoie 7.

$u_1 = 3$ et, pour tout entier naturel $n \geqslant 1$, $u_{n+1} = 0,9u_n + 1,3$.

1. Calculer u_2 et u_3 et proposer une interprétation dans le contexte de l'exercice.

2. Montrer par récurrence que pour tout entier naturel $n \geqslant 1$:

$$u_n = 13 - \frac{100}{9} \times 0,9^n.$$

3. En déduire que la suite (u_n) est croissante.

4. On considère le programme ci-contre, écrit en langage Python.

Déterminer la valeur renvoyée par la saisie de seuil(8.5) et l'interpréter dans le contexte de l'exercice.

```
def seuil(p) :
    n=1
    u=3
    while u<=p :
        n=n+1
        u=0.9*u+1.3
    return n
```

4. On donne ci-contre la fonction seuil, écrite en langage Python.

a. Qu'observe-t-on si on saisit seuil(0.4) ?

b. Déterminer la valeur renvoyée par la saisie de seuil(0.35).

Interpréter cette valeur dans le contexte de l'exercice.

```
def seuil(a) :
    v=0.1
    n=0
    while v<a :
        v=1.6*v-1.6*v*v
        n=n+1
    return n
```

---

5. On considère la fonction Python ci-dessous. Que renvoie-t-elle?

```
def seuil (x) :
    n=1
    while 1-0.96**n < x :
      n = n + 1
    return n
```

a. Le plus petit nombre n tel que la probabilité de tirer au moins une pièce défectueuse soit supérieure ou égale à x.

b. Le plus petit nombre n tel que la probabilité de ne tirer aucune pièce défectueuse soit supérieure ou égale à x.

c. Le plus grand nombre n tel que la probabilité de ne tirer que des pièces défectueuses soit supérieure ou égale à x.

d. Le plus grand nombre n tel que la probabilité de ne tirer aucune pièce défectueuse soit supérieure ou égale à x.

---

4. On donne une fonction écrite en langage Python :

```
def mystere(seuil) :
    n=0
    u=400
    while u <= seuil :
        n = n+1
        u = 0.9*u+60
    return n
```

Quelle valeur obtient-on en tapant dans la console de Python : mystere (500) ?

1. **a.** Vérifier que $v_0 = 6 \times 10^{21}$.

   b. Expliquer que, pour tout nombre entier naturel n, on a

$$v_{n+1} = 0,995 v_n + 1,5 \times 10^{19}.$$

2. **a.** Démontrer, par récurrence sur n, que $0 \leqslant v_{n+1} \leqslant v_n$.

   b. En déduire que la suite $(v_n)_{n\in\mathbb{N}}$ est convergente.

3. On considère la suite $(u_n)_{n\in\mathbb{N}}$ définie, pour tout entier naturel n, par :

$$u_n = v_n - 3 \times 10^{21}.$$

   a. Montrer que la suite $(u_n)_{n\in\mathbb{N}}$ est géométrique de raison 0,995.

   b. En déduire que, pour tout entier naturel n, $v_n = 3 \times 10^{21}\,(0,995^n + 1)$.

   c. En déduire la limite de la suite $(v_n)_{n\in\mathbb{N}}$ et interpréter le résultat dans le contexte de l'exercice.

4. Déterminer, par le calcul, au bout de combien de jours le nombre de noyaux de polonium sera inférieur à $4,5 \times 10^{21}$. Justifier la réponse.

5. On souhaite disposer de la liste des termes de la suite $(v_n)_{n\in\mathbb{N}}$.
   Pour cela, on utilise une fonction appelée noyaux programmée en langage Python et retranscrite partiellement ci-après.

```
1  def noyaux (n) :
2      V =6*10**21
3      L=[V]
4      for k in range (n) :
5          V= ...
6          L.append(V)
7      return L
```

   a. À la lecture des questions précédentes, proposer deux solutions différentes pour compléter la ligne 5 de la fonction noyaux afin qu'elle réponde au problème.

   b. Pour quelle valeur de l'entier n la commande noyaux(n) renverra-t-elle les relevés quotidiens du nombre de noyaux contenus dans l'échantillon de polonium pendant 52 semaines d'étude?

5. Voici un script, écrit en langage Python, relatif aux suites étudiées dans cette partie :

```
1  def heron(n):
2    L=5
3    ℓ=2.2
4    for i in range(n):
5      L = (L+ℓ) / 2
6      ℓ = 11 / L
7    return round(ℓ,6), round(L,6)
```

On rappelle que la fonction Python `round(x,k)` renvoie une version arrondie du nombre x avec k décimales.

a. Si l'utilisateur tape `heron(3)` dans une console d'exécution Python, qu'obtient-il comme valeurs de sortie pour ℓ et L?

b. Donner une interprétation de ces deux valeurs.

EXERCICE 3 — 5 points

On considère la suite (u_n) définie par : $\begin{cases} u_1 = \dfrac{1}{\mathrm{e}} \\ u_{n+1} = \dfrac{1}{\mathrm{e}}\left(1+\dfrac{1}{n}\right)u_n \text{ pour tout entier } n \geqslant 1. \end{cases}$

1. Calculer les valeurs exactes de u_2 et u_3. On détaillera les calculs.

2. On considère une fonction écrite en langage Python qui, pour un entier naturel n donné, affiche le terme u_n. Compléter les lignes L_2 et L_4 de ce programme.

L_1	`def suite(n):`
L_2	`  ..................`
L_3	`  for i in range(1, n):`
L_4	`    u=..................`
L_5	`return u`

3. On admet que tous les termes de la suite (u_n) sont strictement positifs.

a. Montrer que pour tout entier naturel n non nul, on a : $1+\dfrac{1}{n} \leqslant \mathrm{e}$.

b. En déduire que la suite (u_n) est décroissante.

c. La suite (u_n) est-elle convergente? Justifier votre réponse.

4. **a.** Montrer par récurrence que pour tout entier naturel non nul, on a : $u_n = \dfrac{n}{\mathrm{e}^n}$.

b. En déduire, si elle existe, la limite de la suite (u_n).

On considère la suite (u_n) définie par $u_0 = 3$ et, pour tout entier naturel n,

$$u_{n+1} = \frac{1}{2}u_n + \frac{1}{2}n + 1.$$

Partie A

Cette partie est un questionnaire à choix multiples.
Pour chacune des questions suivantes, une seule des quatre réponses proposées est exacte.
Pour répondre, indiquer sur la copie le numéro de la question et la lettre de la réponse choisie.
Aucune justification n'est demandée.
Une réponse fausse, une absence de réponse, ou une réponse multiple, ne rapporte ni n'enlève de point.

1. La valeur de u_2 est égale à :

a. $\frac{11}{4}$ **b.** $\frac{13}{2}$
c. 3,5 **b.** 2,7

2. La suite (v_n) définie, pour tout entier naturel n, par $v_n = u_n - n$ est :

a. arithmétique de raison $\frac{1}{2}$ **b.** géométrique de raison $\frac{1}{2}$
c. constante. **d.** ni arithmétique, ni géométrique.

3. On considère la fonction ci-dessous, écrite de manière incomplète en langage Python.

n désigne un entier naturel non nul.
On rappelle qu'en langage Python « i in range (n) » signifie que i varie de 0 à $n-1$.

```
1  def terme (n)
2      U=3
3      for i in range(n) :
4  ..........................
5      return U
```

Pour que terme (n) renvoie la valeur de u_n, on peut compléter la ligne 4 par :

a. U = U/2 + (i+1)/2+1 **b.** U = U/2 + n/2 + 1
c. U = U/2 + (i−1)/2+1 **d.** U = U/2 + i/2 + 1

3. Montrer que la suite (a_n) vérifie la relation suivante pour tout entier naturel n, on a :

$$a_{n+1} = 0,75a_n + 300.$$

4. **a.** Démontrer par récurrence que pour tout entier naturel n, on a :

$$1200 \leqslant a_{n+1} \leqslant a_n \leqslant 1700.$$

b. En déduire que la suite (a_n) converge.

5. Soit (v_n) la suite définie pour tout entier naturel n par $v_n = a_n - 1200$.
   a. Démontrer que la suite (v_n) est géométrique.
   b. Exprimer v_n en fonction de n.
   c. En déduire que pour tout entier naturel n, $a_n = 500 \times 0,75^n + 1200$.

6. **a.** Déterminer la limite de la suite (a_n).
   b. Interpréter le résultat de la question précédente dans le contexte de l'exercice.

7. **a.** Recopier et compléter le programme Python ci-dessous afin qu'il renvoie la plus petite valeur de n à partir de laquelle le nombre de membres du club A est strictement inférieur à 1 280.

```
def seuil() :
    n = 0
    A = 1 700
    while ... :
        n=n+1
        A = ...
    return...
```

b. Déterminer la valeur renvoyée lorsqu'on appelle la fonction seuil.

On considère la suite (u_n) définie par $u_0 = 8$ et, pour tout entier naturel n,

$$u_{n+1} = \frac{6u_n + 2}{u_n + 5}.$$

1. Calculer u_1.

2. Soit f la fonction définie sur l'intervalle $[0\,;\,+\infty[$ par :

$$f(x) = \frac{6x + 2}{x + 5}.$$

Ainsi, pour tout entier naturel n, on a : $u_{n+1} = f(u_n)$.

a. Démontrer que la fonction f est strictement croissante sur l'intervalle $[0\,;\,+\infty[$.
En déduire que pour tout réel $x > 2$, on a $f(x) > 2$.

b. Démontrer par récurrence que, pour tout entier naturel n, on a $u_n > 2$.

3. On admet que, pour tout entier naturel n, on a :

$$u_{n+1} - u_n = \frac{(2 - u_n)(u_n + 1)}{u_n + 5}.$$

a. Démontrer que la suite (u_n) est décroissante.

b. En déduire que la suite (u_n) est convergente.

4. On définit la suite (v_n) pour tout entier naturel par :

$$v_n = \frac{u_n - 2}{u_n + 1}.$$

a. Calculer v_0.

b. Démontrer que (v_n) est une suite géométrique de raison $\frac{4}{7}$.

c. Déterminer, en justifiant, la limite de (v_n).
En déduire la limite de (u_n).

5. On considère la fonction Python seuil ci-contre, où A est un nombre réel strictement plus grand que 2.

Donner, sans justification, la valeur renvoyée par la commande seuil (2.001) puis interpréter cette valeur dans le contexte de l'exercice.

```
def seuil (A) :
    n = 0
    u = 8
    while u > A :
        u = (6*u + 2)/(u + 5)
        n = n + 1
    return n
```

3. On considère la fonction ci-dessous, écrite de manière incomplète en langage Python.

n désigne un entier naturel non nul.
On rappelle qu'en langage Python « i in range (n) » signifie que i varie de 0 à $n-1$.

```
1  def terme (n)
2      U=3
3      for i in range(n) :
4      .....
5      return U
```

Pour que terme (n) renvoie la valeur de u_n, on peut compléter la ligne 4 par :

a. U = U/2 + (i+1)/2+1
b. U = U/2 + n/2 + 1
c. U = U/2 + (i−1)/2+1
d. U = U/2 + i/2 + 1

On considère la suite (u_n) définie par $u_0 = 3$ et, pour tout entier naturel n, par :

$$u_{n+1} = 5u_n - 4n - 3.$$

1. **a.** Démontrer que $u_1 = 12$.
b. Déterminer u_2 en détaillant le calcul.
c. À l'aide de la calculatrice, conjecturer le sens de variation ainsi que la limite de la suite (u_n).

2. **a.** Démontrer par récurrence que, pour tout entier naturel n, on a :

$$u_n \geqslant n+1.$$

b. En déduire la limite de la suite (u_n).

3. On considère la suite (v_n) définie pour tout entier naturel n par :

$$v_n = u_n - n - 1.$$

a. Démontrer que la suite (v_n) est géométrique.
Donner sa raison et son premier terme v_0.
b. En déduire, pour tout entier naturel n, l'expression de v_n en fonction de n.
c. En déduire que pour tout entier naturel n :

$$u_n = 2 \times 5^n + n + 1.$$

d. En déduire le sens de variation de la suite (u_n).

4. On considère la fonction ci-contre, écrite de manière incomplète en langage Python et destinée à renvoyer le plus petit entier naturel n tel que $u_n \geqslant 10^7$.

a. Recopier le programme et compléter les deux instructions manquantes.
b. Quelle est la valeur renvoyée par cette fonction ?

```
def suite() :
    u = 3
    n = 0
    while ... :
        u = ...
        n = n + 1
    return n
```

On considère la suite (u_n) telle que $u_0 = 0$ et pour tout entier naturel n :

$$u_{n+1} = \frac{-u_n - 4}{u_n + 3}.$$

On admet que u_n est défini pour tout entier naturel n.

1. Calculer les valeurs exactes de u_1 et u_2.
2. On considère la fonction `terme` ci-dessous écrite de manière incomplète en langage Python :

```
def terme (n) :
    u = ...
    for i in range(n):
       u = ...
    return(u)
```

On rappelle qu'en langage Python, « `i in range (n)` » signifie que i varie de 0 à $n-1$.

Recopier et compléter le cadre ci-dessus de sorte que, pour tout entier naturel n, l'instruction `terme (n)` renvoie la valeur de u_n.

Pour les questions **3.** et **4.**, on considère la suite (u_n) définie sur $\mathbb{N}$ par :

$$u_0 = 15 \quad \text{et pour tout entier naturel } n \quad : u_{n+1} = 1,2u_n + 12.$$

3. La fonction Python suivante, dont la ligne 4 est incomplète, doit renvoyer la plus petite valeur de l'entier n telle que $u_n > 10000$.

```
def seuil() :
    n=0
    u=15
    while ......:
      n=n+1
      u=1,2*u+12
    return(n)
```

À la ligne 4, on complète par :

a. u$\leqslant 10000$; **b.** u $= 10000$ **c.** u> 10000; **d.** $n \leqslant 10000$.

$$f(x) - x = \frac{3}{4}(x-2)^2.$$

On considère la suite (u_n) définie par un réel u_0 et pour tout entier naturel n :

$$u_{n+1} = f(u_n).$$

On a donc, pour tout entier naturel n,

$$u_{n+1} = \frac{3}{4}u_n^2 - 2u_n + 3.$$

4. Étude du cas : $\frac{4}{3} \leqslant u_0 \leqslant 2$.

a. Démontrer par récurrence que, pour tout entier naturel n,

$$u_n \leqslant u_{n+1} \leqslant 2.$$

b. En déduire que la suite (u_n) est convergente.

c. Prouver que la limite de la suite est égale à 2.

5. Étude du cas particulier : $u_0 = 3$.

On admet que dans ce cas la suite (u_n) tend vers $+\infty$.

Recopier et compléter la fonction « `seuil` » suivante écrite en Python, afin qu'elle renvoie la plus petite valeur de n telle que u_n soit supérieur ou égal à 100.

```
def seuil() :
  u = 3
  n = 0
  while ...
      u = ...
      n = ...
  return n
```

6. Étude du cas : $u_0 > 2$.

À l'aide d'un raisonnement par l'absurde, montrer que (u_n) n'est pas convergente.

On considère la fonction f définie sur l'ensemble $]0\,;\,+\infty[$ par

$$f(x) = 1 + x^2 - 2x^2\ln(x).$$

On admet que f est dérivable sur l'intervalle et on note f' sa fonction dérivée

1. Justifier que $\lim\limits_{x\to 0} f(x) = 1$ et, en remarquant que $f(x) = 1 + x^2(1 - 2\ln(x))$, justifier que $\lim\limits_{x\to+\infty} f(x) = -\infty$.
2. Montrer que pour tout réel x de l'intervalle $]0\,;\,+\infty[$, $\quad f'(x) = -4x\ln(x)$.
3. Étudier le signe de $f'(x)$ sur l'intervalle $]0\,;\,+\infty[$, puis dresser le tableau de variation de la fonction sur l'intervalle $]0\,;\,+\infty[$.
4. Démontrer que l'équation $f(x) = 0$ admet une unique solution α dans l'intervalle $[1\,;\,+\infty[$ et que $\alpha \in [1\,;\,e]$.

On admet dans la suite de l'exercice, que l'équation $f(x) = 0$ n'admet pas de solution sur l'intervalle $]0; 1]$.

5. On donne la fonction ci-dessous écrit en Python. L'instruction *from lycee import ** permet d'accéder à la fonction ln.

```
from lycee import *

def f(x) :
    return 1+x**2-2*x**2*ln(x)

def dichotomie(p) :
    a=1
    b=2.7
    while b-a > 10**(-p) :
        if f(a)*f((a+b)/2) < 0 :
        |  b = (a+b)/2
        else :
        |  a=(a+b)/2
          return (a,b)
```

Il écrit dans la console d'exécution :

```
>>> dichotomie(1)
```

Parmi les quatre propositions ci-dessous, recopier celle affichée par l'instruction précédente? Justifier votre réponse (on pourra procéder par élimination)

Proposition A : (1.75, 1.9031250000000002)
Proposition B : (1.85, 1.9031250000000002)
Proposition C : (2.75, 2.9031250000000002)
Proposition D : (2.85, 2.9031250000000002)

Exercice 3 **5 points**

Soit la suite (u_n) définie par $u_0 = 0$ et, pour tout $n \in \mathbb{N}$,

$$u_{n+1} = 5u_n - 8n + 6.$$

1. Calculer u_1 et u_2.
2. Soit n un entier naturel.
   Recopier et compléter la fonction `suite_u` d'argument `n` ci-dessous, écrite en langage Python, afin qu'elle retourne la valeur de u_n.

```
def suite_u(n) :
    u = ...
    for i in range(1,n+1) :
    |    u= ...
    return u
```

3. **a.** Démontrer par récurrence que, pour tout $n \in \mathbb{N}$, $\quad u_n \geqslant 2n$.
   b. En déduire la limite de la suite (u_n).
   c. Soit $p \in \mathbb{N}^*$. Pourquoi peut-on affirmer qu'il existe au moins un entier n_0 tel que, pour tout entier naturel n vérifiant, $n \geqslant n_0$, $u_n \geqslant 10^p$?
4. Démontrer que la suite (u_n) est croissante.
5. On considère la suite (v_n), définie pour tout $n \in \mathbb{N}$, par $v_n = u_n - 2n + 1$.
   a. En dessous de la fonction `suite_u` précédente, on a écrit la fonction `suite_v` ci-dessous :

```
def suite_v(n):
    L =[]
    for i in range(n+1) :
    |    L.append(suite_u(i)-2*i+1)
    return L
```

   La commande « L.append »permet de rajouter, en dernière position, un élément dans la liste L.
   Lorsqu'on saisit `suite_v(5)` dans la console, on obtient l'affichage suivant :

```
>>> suite_v(5)
[1, 5, 25, 125, 625, 3125]
```

   Conjecturer, pour tout entier naturel n, l'expression de v_{n+1} en fonction de v_n.
   Démontrer cette conjecture.
   b. En déduire, pour tout entier naturel n, la forme explicite de u_n en fonction de n.