Exercice 1

Partie A : Étude de la suite $(u_n)_{n\in\mathbf{N}}$

On s'intéresse à la suite récurrente $(u_n)_{n\in\mathbf{N}}$ définie par $u_0 = 1$ et

$$\forall n \in \mathbf{N}, \quad u_{n+1} = u_n \, \mathrm{e}^{1/u_n}.$$

1. (a) Montrer que $u_n > 0$ pour tout $n \in \mathbf{N}$.
   (b) Donner le sens de variation de la suite $(u_n)_{n\in\mathbf{N}}$.
   (c) Démontrer, en raisonnant par l'absurde, que $(u_n)_{n\in\mathbf{N}}$ admet $+\infty$ comme limite.
2. Recopier et compléter le programme Python ci–dessous de sorte qu'il affiche le premier entier $n \in \mathbf{N}$ tel que $u_n \geqslant 10^6$.

```
import numpy as np
u = 1
n = 0
while ... :
    u = ...
    n = ...
print(...)
```

Partie B : Étude de la fonction f

On considère la fonction f définie sur $]0, +\infty[$ par :

$$\forall x > 0, \quad f(x) = x \, \mathrm{e}^{1/x}.$$

On note $\mathscr{C}_f$ la courbe de f dans le plan muni d'un repère orthonormé.

3. Calculer les limites de f en $+\infty$ et en 0.
4. Dresser le tableau de variation de f sur $]0, +\infty[$.
5. Soit $x > 0$.
   (a) Justifier la convergence de la série $\sum\limits_{k\geqslant 0} \dfrac{x^{-k}}{k!}$ et calculer sa somme.
   (b) En déduire que :
   $$f(x) = x + 1 + \frac{1}{x} \sum_{k=2}^{+\infty} \frac{x^{2-k}}{k!}.$$
6. Soit $x \geqslant 1$.
   (a) Établir séparément les inégalités suivantes :
   $$\frac{1}{2} \leqslant \sum_{k=2}^{+\infty} \frac{x^{2-k}}{k!} \leqslant \mathrm{e}.$$
   (b) En déduire que :
   $$(*) \qquad \frac{1}{2x} \leqslant f(x) - (x+1) \leqslant \frac{\mathrm{e}}{x}.$$
7. Montrer que $f(x) = x + 1 + o(1)$ au voisinage de $+\infty$.
8. Représenter sur un même dessin la courbe $\mathscr{C}_f$ et la droite d'équation $y = x + 1$.

Partie C : Comportement asymptotique de la suite $(u_n)_{n\in\mathbf{N}}$

9. (a) Montrer que, pour tout entier $k \in \mathbf{N}$, $\quad \ln(u_{k+1}) - \ln(u_k) = \dfrac{1}{u_k}$.

   (b) En déduire que, pour tout entier $n \in \mathbf{N}^*$, $\quad \ln(u_n) \ = \ \displaystyle\sum_{k=0}^{n-1} \frac{1}{u_k}$.

10. (a) À l'aide de l'encadrement $(*)$ montrer que, pour tout $k \in \mathbf{N}$,

$$1 + \frac{1}{2u_k} \ \leqslant \ u_{k+1} - u_k \ \leqslant \ 1 + \frac{\mathrm{e}}{u_k}.$$

   (b) Soit $n \in \mathbf{N}^*$, établir :

$$n + \frac{1}{2}\sum_{k=0}^{n-1} \frac{1}{u_k} \ \leqslant \ u_n - 1 \ \leqslant \ n + \mathrm{e}\sum_{k=0}^{n-1} \frac{1}{u_k},$$

   puis

$$1 + \frac{1}{2}\ln(u_n) \ \leqslant \ u_n - n \ \leqslant \ 1 + \mathrm{e}\ln(u_n).$$

11. (a) Justifier que : $\displaystyle\lim_{n\to+\infty} \frac{\ln(u_n)}{u_n} = 0$.

   (b) En déduire un équivalent simple de u_n lorsque n tend vers $+\infty$.

12. Déterminer un équivalent simple de $\displaystyle\sum_{k=0}^{n-1} \frac{1}{u_k}$ lorsque n tend vers $+\infty$.

Exercice 2

Les deux parties de cet exercice sont indépendantes l'une de l'autre.

Partie A : Réduction simultanée et spectre

Soit $\mathscr{M}_3(\mathbf{R})$ l'espace vectoriel des matrices carrées d'ordre trois à coefficients réels. On pose :

$$I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad J = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad \text{et} \quad K = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix},$$

et on considère $\mathscr{E} = \mathrm{Vect}(I, J, K)$ le sous-espace vectoriel de $\mathscr{M}_3(\mathbf{R})$ engendré par les matrices I, J et K.

1. Montrer que (I, J, K) est une base de $\mathscr{E}$, en déduire la dimension de $\mathscr{E}$.

2. Justifier sans calcul que les matrices J et K sont diagonalisables.

3. (a) Exprimer la matrice J^3 comme un multiple de J.

   (b) En déduire que les valeurs propres de J appartiennent à l'ensemble $\{-\sqrt{2}, 0, \sqrt{2}\}$.

On pose $U_1 = \begin{pmatrix} \sqrt{2} \\ 1 \\ 1 \end{pmatrix}$ et $U_2 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$.

4. (a) Vérifier que U_1 et U_2 sont des vecteurs propres de J.

   (b) Déterminer un vecteur propre U_3 de J associé à la valeur propre $-\sqrt{2}$.

5. (a) Justifier que (U_1, U_2, U_3) est une base de $\mathscr{M}_{3,1}(\mathbf{R})$.

(b) Donner une matrice inversible P de $\mathscr{M}_3(\mathbf{R})$ telle que :

$$P^{-1}JP = \begin{pmatrix} \sqrt{2} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -\sqrt{2} \end{pmatrix}.$$

6. (a) Montrer que (U_1, U_2, U_3) est aussi une base de vecteurs propres de K.

   (b) Déterminer la matrice $P^{-1}KP$.

7. Soit M une matrice de $\mathscr{E}$ de coordonnées $(a, b, c) \in \mathbf{R}^3$ dans la base (I, J, K).

   (a) Exprimer la matrice $P^{-1}MP$ sous la forme d'un tableau de nombres dépendant de a, b et c.

   (b) En déduire les valeurs propres de M.

8. On considère l'application linéaire $s\colon \mathscr{E} \to \mathbf{R}^3$ définie par :

$$s(M) = s(aI + bJ + cK) = \left(a + b\sqrt{2} + c, a - c, a - b\sqrt{2} + c\right)$$

   pour toute matrice $M = aI + bJ + cK$ avec $(a, b, c) \in \mathbf{R}^3$.

   (a) Donner la matrice S de s relativement à la base (I, J, K) de $\mathscr{E}$ et à la base canonique de $\mathbf{R}^3$.

   (b) Montrer que l'application linéaire s est bijective.

Partie B : Un algorithme de coloration des graphes

Soit $n \geqslant 1$ un entier, on considère un graphe non orienté G donné par sa matrice d'adjacence $A \in \mathscr{M}_n(\mathbf{R})$. On note $\mathscr{S} = \{s_0, \ldots, s_{n-1}\}$ l'ensemble des sommets de G, dans les programmes informatiques on confondra un sommet s_i avec son numéro i. On dit que deux sommets sont *voisins* s'ils sont distincts et reliés par une arête.

Une *coloration* de G est une application $c\colon \mathscr{S} \to \mathbf{N}$ telle que $c(s_i) \neq c(s_j)$ si les sommets s_i et s_j sont voisins. Dans cette définition, $\mathbf{N}$ représente l'ensemble des « couleurs » disponibles, la coloration c attribue à chaque sommet une « couleur » de sorte que deux sommets voisins soient de « couleurs » différentes.

Le graphe G admet la *coloration triviale* donnée par $c(s_i) = i$ pour tout $i \in [\![0, n-1]\!]$, il peut cependant admettre une coloration nécessitant moins de n « couleurs ». Ainsi, le graphe à cinq sommets ci–dessous admet la coloration à trois « couleurs » définie par : $c(s_0) = 0$, $c(s_1) = 1$, $c(s_2) = 0$, $c(s_3) = 1$, $c(s_4) = 2$.

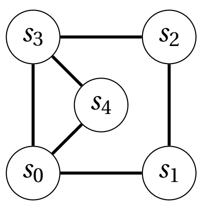

Figure 1 : Un graphe d'ordre cinq

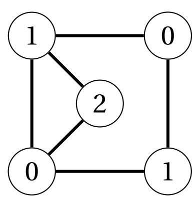

Figure 2 : Le graphe colorié avec trois « couleurs » (0, 1 et 2)

Les questions suivantes ont pour but de réaliser un programme Python qui renvoie une coloration d'un graphe G quelconque, en essayant de minimiser le nombre de couleurs utilisées. On commence par rédiger deux fonctions auxiliaires, « `voisins` » et « `min_ext` », qui serviront pour la fonction finale « `coloration` ». On suppose que la matrice d'adjacence A de G est définie à l'aide de la commande « `np.array` ».

9. Recopier et compléter le programme Python ci–dessous de manière à ce qu'il définisse une fonction « `voisins` », prenant en arguments la matrice d'adjacence A et un entier $i \in [\![0, n-1]\!]$, et renvoyant la liste des sommets voisins de s_i.

```
def voisins(A,i):
    n = len(A[i])
    V = []
    for j in range(n):
        if j!= i and ... :
            V.append(...)
    return(V)
```

10. Rédiger en Python une fonction « `min_ext` » qui prend en argument une liste d'entiers naturels L, et qui renvoie le plus petit entier naturel n'appartenant pas à L (par exemple, si $L = [1,0,3,]$ alors la commande « `min_ext(L)` » renvoie 2). On pourra transcrire en langage Python l'algorithme suivant :

> On affecte à une variable m la valeur 0.
>
> Tant que m appartient à la liste L :
>
> | On augmente de 1 la valeur de m.
>
> On renvoie m.

11. À l'aide des fonctions introduites précédemment on rédige maintenant une fonction « `coloration` » prenant en argument la matrice d'adjacence $A \in \mathscr{M}_n(\mathbf{R})$ d'un graphe G, et renvoyant une coloration de G sous la forme d'une liste d'entiers $C = [C_0, \dots, C_{n-1}]$, où C_i désigne la « couleur » du sommet s_i pour tout $i \in [\![0, n-1]\!]$.

    On construit cette fonction selon l'algorithme « glouton » ci–dessous :

> On affecte à la variable n le nombre de sommets de G.
>
> On affecte à la variable C la liste $[0, 1, \dots, n-1]$.
>
> Pour i allant de 1 à $n-1$:
>
> | On affecte à la variable « `C_voisins` » la liste des « couleurs » des sommets voisins de s_i
> | On affecte à C_i le plus petit entier naturel qui n'est pas élément de la liste « `C_voisins` ».
>
> On renvoie la liste C.

Recopier et compléter la fonction « `coloration` » ci–dessous.

```
def coloration(A):
    n = len(A[0])
    C = ...
    for i in range(1,n):
        C_voisins = [ ... for j in ... ]
        C[i] = min_ext(...)
    return(C)
```

12. On note A la matrice d'adjacence du graphe G représenté en figure 3 ci–contre.

    (a) Donner la liste obtenue en exécutant la commande « `coloration(A)` ».

    (b) Le graphe G admet–il une coloration à trois couleurs ? Si oui, exhiber une telle coloration.

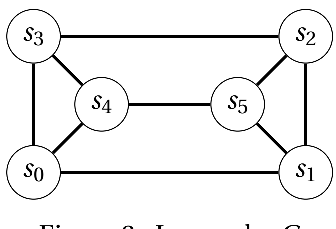

Figure 3 : Le graphe G

Exercice 3

Les parties B, C et D de cet exercice sont indépendantes les unes des autres.
Toutes les variables aléatoires sont supposées définies sur un même espace probabilisé $(\Omega, \mathscr{A}, P)$.

Partie A : La variable aléatoire V

Soit U une variable aléatoire suivant la loi uniforme sur $]0,1]$, on note V la variable aléatoire définie par :

$$V = \frac{1}{\sqrt{U}}.$$

1. (a) Justifier que V est à valeurs dans $[1, +\infty[$.
   (b) Montrer que la fonction de répartition de V est donnée par :
   $$F_V(x) = \begin{cases} 1 - \dfrac{1}{x^2} & \text{si } x \geqslant 1, \\ 0 & \text{si } x < 1. \end{cases}$$
   (c) En déduire que V est une variable aléatoire à densité, et donner une densité f_V de V.
2. Déterminer si V admet une espérance et une variance, calculer leurs valeurs éventuelles.

La variable aléatoire V suit une *loi de Pareto*, les compagnies d'assurance utilisent cette loi pour modéliser les montants des sinistres. Afin d'établir des prévisions, un actuaire étudie une suite $(V_i)_{i\geqslant 1}$ de variables aléatoires mutuellement indépendantes et suivant la même loi que V, la variable aléatoire V_i représente le coût du i–ième sinistre survenu à partir d'un instant donné.

Partie B : Loi du sinistre le plus coûteux

Pour tout entier $n \geqslant 1$ on définit une variable aléatoire M_n en posant :

$$M_n = \max(V_1, \dots, V_n).$$

On note F_n la fonction de répartition de M_n.

3. (a) Montrer que $F_n = (F_V)^n$ pour tout entier $n \geqslant 1$.
   (b) Calculer la limite $\lim\limits_{n\to+\infty} F_n(x)$ pour tout $x \in \mathbf{R}$.
   (c) Justifier que la suite $(M_n)_{n\geqslant 1}$ ne converge en loi vers aucune variable aléatoire.

On considère une variable aléatoire W dont la fonction répartition F_W est définie par :

$$F_W(x) = \begin{cases} \mathrm{e}^{-\frac{1}{x^2}} & \text{si } x > 0, \\ 0 & \text{si } x \leqslant 0. \end{cases}.$$

Pour tout entier $n \geqslant 1$, on note G_n la fonction de répartition de la variable aléatoire $\dfrac{M_n}{\sqrt{n}}$.

4. (a) Montrer que $\lim\limits_{n\to+\infty} G_n(x) = \mathrm{e}^{-\frac{1}{x^2}}$ pour tout $x > 0$.
   (b) Conclure quant à la convergence en loi de la suite $\left(\dfrac{M_n}{\sqrt{n}}\right)_{n\geqslant 1}$.

Partie C : Manipulation d'une base de données

La compagnie d'assurance tient à jour une table « `sinistres` » contenant des informations sur tous les sinistres qu'elle a indemnisés entre les années 2000 et 2024. Les attributs (colonnes) de cette table sont :

- `id` (de type `INTEGER`) : numéro d'identification du sinistre,
- `annee` (de type `INTEGER`) : année durant laquelle est survenu le sinistre,
- `mois` (de type `TEXT`) : mois durant lequel est survenu le sinistre (on écrit le mois en minuscules),
- `montant` (de type `INTEGER`) : montant de l'indemnisation versée à l'assuré (en euros).

5. Rédiger une requête SQL permettant d'afficher :
   (a) La liste des montants d'indemnisation des sinistres de l'année 2024.
   (b) Le mois et l'année de tous les sinistres dont le montant d'indemnisation dépasse un million.
6. Le sinistre numéro 7652 s'est produit en avril 2025 et a été indemnisé à hauteur de 1540 euros. Rédiger une requête SQL ajoutant à la table « `sinistre` » une ligne correspondant à ce sinistre.

Partie D : Nombre de sinistres graves

On rappelle que $(V_i)_{i\geqslant 1}$ est une suite de variables aléatoires mutuellement indépendantes suivant toutes la même loi que V (voir partie A). On suppose que le nombre de sinistres se produisant au cours d'une année est donné par une variable aléatoire N suivant une loi de Poisson de paramètre $\lambda > 0$. On s'intéresse au nombre de sinistres dont le coût dépasse un certain montant $A > 1$. On note ainsi T la variable aléatoire égale au nombre d'éléments de $\{V_1, \dots, V_N\}$ prenant une valeur supérieure à A, formellement :

$$\forall \omega \in \Omega, \quad T(\omega) = |\{i \in [\![1, N(\omega)]\!] \; ; \; V_i(\omega) > A\}|,$$

où la notation $|\cdot|$ désigne le cardinal.

7. Exprimer $P(N = n)$ pour tout $n \in N(\Omega)$.
8. Quel est l'ensemble $T(\Omega)$ des valeurs prises par T ?
9. Soit $n \in \mathbf{N}^*$.
   (a) Justifier que la loi conditionnelle de T sachant $(N = n)$ est la loi binomiale $\mathscr{B}\left(n, \frac{1}{A^2}\right)$.
   (b) Donner la valeur de $P_{(N=n)}(T = k)$ pour tout $k \in \mathbf{N}$, vous distinguerez les cas $k \leqslant n$ et $k > n$.
10. Calculer $P(T = k)$ pour tout $k \in \mathbf{N}$, puis reconnaître la loi de T.
11. En moyenne, combien de sinistres avec un coût supérieur à A surviennent en un an ?

Fin de l'énoncé