

Toutes les réponses doivent être regroupées dans un unique fichier Python exécutable.
Chaque fonction doit être testée par au moins un appel visible.

A vous de choisir l'ordre dans lequel vous allez faire les questions .

(Vous devez tous progresser sur ce sujet : connaître vos points forts et points faibles)

Vous devez répondre aux questions et non pas faire des copier/coller des fichiers vus en classe.

1. Tracer une courbe et trouver la solution d'une équation.

- (a) Tracer avec le module `numpy` et `matplotlib.pyplot` sur le même repère et sur $[0, 2]$ les courbes d'équation $y = \cos(x)$ et $y = x$.
- (b) Tracer sans le module `numpy` (mais avec `math` et `matplotlib.pyplot`) sur le même repère et sur $[0, 2]$ les courbes d'équation $y = \cos(x)$ et $y = x$.
- (c) Déterminer une solution approchée à 10^{-3} près de $\cos(x) = x$ sur l'intervalle $[0, 2]$.

Vous préciserez la méthode utilisée.

2. Sur des listes.

Ecrire une fonction `amplitude(L)` qui prend en entrée une liste de nombres `L` et qui renvoie la différence entre sa valeur maximale et sa valeur minimale.

On ne parcourera qu'une fois la liste `L` et on n'utilisera pas les fonctions natives `max` et `min`

3. Tester une propriété.

On dit qu'une matrice carrée réelle $M = (m_{i,j})$ est *diagonale à dominante stricte par lignes* si :

$$\textbf{1 } M \text{ est carrée et } \textbf{2 } \text{ Pour tout indice } i, \quad |m_{i,i}| > \sum_{j \neq i} |m_{i,j}|$$

Une telle matrice est automatiquement inversible.

Ecrire une fonction Python `dominante_lignes(M)` qui renvoie `True` si une matrice M vérifie la condition de dominance stricte par lignes, et `False` sinon.

On supposera que `M` est une donnée sous forme d'une liste de listes.

4. Sur des chaînes de caractères.

Ecrire une fonction `Palindrome(T)` qui prend en entrée une chaîne de caractères `T` et qui renvoie `True` si `T` est un palindrome et `False` sinon.

5. Calculer une intégrale.

Calculer en appliquant la méthode des rectangles à gauche une valeur approchée de l'intégrale :

$$\int_0^2 \sin(x^2) dx$$

6. Estimer une espérance ou une loi par une série de simulation

On considère l'instruction `int(4*rd.random()*2+1)` qui permet de simuler une variable aléatoire discrète X .

On importera `random` avec l'instruction : `import random as rd`

- (a) Estimer l'espérance de X à l'aide d'une série de simulations.
- (b) Estimer la loi de X à l'aide d'une série de simulations.

7. Tracer les termes d'une suite.

Ecrire une fonction Python qui pour un entier n renvoie la liste des termes x_k (à 10^{-3} près) pour k allant de 3 à n de la suite (x_n) définie par : pour chaque entier $n \geq 3$, x_n est l'unique solution sur $[0, 1]$ de l'équation :

$$x^n + x^2 + 2x - 1 = 0$$

On utilisera l'algorithme de dichotomie.

8. Méthode d'Euler.

Estimer avec la méthode d'Euler le minimum de la solution du problème différentiel suivant :

$$\text{sur } [-2, 3], \quad \begin{cases} y''(t) - y'(t)^2 + 2y(t) = 2 \\ y(0) = 2 \quad \text{et} \quad y'(0) = -1 \end{cases}$$

Commencer par transformer cette équation d'ordre 2 en un système du premier ordre.