Examen : Python 3 pour la science

Consigne :

- Votre fichier/dossier de rendu devra comporter votre nom et prénom sur son nommage (NOM_Prenom.ipny ou NOM_Prenom pour le dossier).
- Un bloc par exercice et chaque bloc doit etre indépendant (import librairie etc...), un fichier par exercice si vous n'utilisez pas jupyter-notebook (exercice_1.py, exercice_2.py etc...).
- Annoter avec # Question N pour localiser vos réponses. Utiliser les commentaires si besoin pour spécifier une réponse littérale si demandé.
- Vous disposez de deux heures et c'est un exercice personelle, tout travail collaboratif sera sanctioné.

Formulaire :

Mathématique : Python

- cos(x) : `np.sin(x)`
- sin(x) : `np.sin(x)`
- e^x : `np.exp(x)`
- π : `np.pi`
- x^y : `x**y`

Exercice 1 : Suite de Fibonacci

Boites à outils à utiliser :

```
import numpy as np
`
```

On se propose d'étudier les 50 premiers éléments de la suite de fibonacci défini par :

$F_0 = 0$

$F_1 = 1$

$F_n = F_{n-1} + F_{n-2}$, pour $n > 2$

1. Créer un **tableau** comprenant 50 zéros, que vous atriburez a une variable `fibo_array` (`np.zeros(N)`)
2. Créer une boucle parcourant ce **tablau** afin d'atribuer les éléments de la suite de fibonacci. (N'oubliez pas l'outils `enumerate(array)` pour récuperer l'indice)
   - Pour l'indice 0 attribué la valeur 0
   - Pour l'indice 1 attribué la valeur 1
   - Pour le reste des indice attribué la valeur `y[i]=y[i-1]+y[i-2]` soit la somme des deux valeurs précédantes du tableau
3. Sur le tableau résultants trouvé et affiché les valeurs pairs.
4. Faites en de même pour les multiples de cinq.
5. La suite de fibonacci est ddéfini par des **entiers**, quel est la **nature** (float, int, bool, lst...) des élements contenus dans votre tableau ?
6. Calculer et afficher le nombre maximum stockable en binanire sur 8-bit pour un entier, de même pour 16-bit, 32-bit,64-bit.
7. Selon ce principe combien de bits sont nécéssaires pour stocker le dernier nombre de notre tableau ?

In [ ]:

Exercice 2 : Courbes et opérations

Boites à outils à utiliser :

```
import numpy as np
import matplotlib.pyplot as plt
`
```

Rappel pour créer une figure :

```
plt.figure() #Instantier l'objet figure

plt.plot(x,y) #Tracer y en fonction de x

plt.title("Titre") #Mettre un titre (optionel)
plt.xlabel("x") #Nommer l'axe des x (optionel)
plt.ylabel("f(x)") #Nommer  l'axe des y (optionel)
plt.xlim(0,10) #fixer la limite d'affichage sur x (optionel)

plt.show() #Afficher la résultante
`
```

Nous allons ici étudier les fonctions :

$f(x) = \frac{3x^2}{50} + sin(x)$ avec $x \in [0, 10[$

$g(y) = 10 + 10cos(\pi . y) * e^{-\frac{y}{12}}$ avec $y \in [5, 25]$

1. Créer un **tableau** allant 0 à 10 (exclu) par pas de 0.1 que vous atriburez à une variable nomée `x_array` .
2. Créer un **tableau** correspondant à **f(x)** appliqué à `x_array` que vous nomeré `f_array` .
3. Créer une **figure** représentant la courbe $f(x) = \frac{3x^2}{50} + sin(x)$ et afficher là.
4. Créer un **tableau** allant 5 à 25 (inclu) par pas de 0.2 que vous atriburez à une variable nomée `y_array` .
5. Créer un **tableau** correspondant à **g(y)** appliqué à `y_array` que vous nomeré `g_array` .
6. Déplacer les deux dernières lignes de votre code pour pouvoir afficher par `plt.plot(x,y)` sur la figure précdément créer, la fonction $g(y) = 10 + 10cos(\pi . y) * e^{-\frac{y}{12}}$.
7. Adapter les limites d'affichages et titre, si nécéssaire.
8. Calculer la résultante des intégrales $\int_a^b f(x) \approx \sum_{i=0}^{N-1} y_i dx_i = \sum_{i=0}^{N-1} y_i * (x_{i+1} - x_i)$ pour les deux fonctions en utilisant la méthode numérique de votre choix. Attention le pas dx est différent pour les deux courbes $f(x)$ et $g(x)$. Afficher le résultat de ces sommes par le biais de `print()` . Discuter la pertinence de votre choix.
9. Afficher les valeurs maximals, minimal et les moyennes pour ces deux courbes.

In [ ]: