Python_Project

July 3, 2025

Burkina Institute of Technology

Année Universitaire 2024-2025

Class : EE, ME, CS

Scientific computing with Python Test

NB : Each group should give a ipynb file which contain :

- The class (EE, ME or CS)
- The names of members
- The script of each exercise

Exercise 1

Définir avec Python la fonction

$$\texttt{f}: \begin{cases} \mathbb{N}^2 \longrightarrow \mathbb{R} \\ (a,b) \longmapsto 333,75\, b^6 + a^2(11a^2b^2 - b^6 - 121\, b^4 - 2) + 5,5\, b^8 + \frac{a}{2b} \end{cases}$$

Avec un logiciel de calcul formel, on trouve `f(77617,33096)=-0.8273960599`. Tester ce résultat en pratique avec Python.

Exercise 2

Les trois fonctions à écrire sont indépendantes et doivent être documentées. On pourra importer certains objets du module `math` évoqués au chapitre 4.

1. Écrire une fonction `volume` qui prend en argument un nombre `r` et renvoie le volume d'une boule de rayon `r`.
2. Écrire une fonction `couronne` qui prend en argument deux nombres `r` et `R` et renvoie l'aire de la surface comprise entre les cercles de rayon `r` et `R` lorsque `r`$\leqslant$`R`, un message d'erreur sinon.
3. Écrire une fonction `distance` qui prend en argument $(x_1, y_1, z_1, x_2, y_2, z_2)$ et renvoie la distance entre les points de coordonnées (x_1, y_1, z_1) et (x_2, y_2, z_2) dans un repère orthonormé.

Exercise 3

Comparer les résultats obtenus en calculant $\sum_{k=1}^{2^{15}} \frac{1}{k^2}$ dans un sens puis dans l'autre. Pour cela, en s'inspirant de la fonction `som_croissante` ci-dessous, écrire la fonction `som_decroissante`, puis calculer la valeur de ces deux fonctions en 2^{15}.

```
def som_croissante(N) :
    S = 0
    for k in range(1,N+1) :
        S += 1/k**2
    return S
```

Exercise 4

On considère la suite numérique (v_n) définie par, pour tout entier naturel n ,

$$\begin{cases} v_0 &= 1 \\ v_{n+1} &= \dfrac{9}{6 - v_n} \end{cases}$$

On souhaite écrire un algorithme affichant, pour un entier naturel n donné, tous les termes de la suite, du rang 0 au rang n.

1. Parmi les trois algorithmes suivants, un seul convient. Préciser lequel en justifiant la réponse.

Algorithme N° 1
Variables : v est un réel i et n sont des entiers naturels **Début de l'algorithme :** Lire n v prend la valeur 1 Pour i variant de 1 à n faire v prend la valeur $\frac{9}{6-v}$ Fin pour Afficher v **Fin algorithme**

Algorithme N° 2
Variables : v est un réel i et n sont des entiers naturels **Début de l'algorithme :** Lire n Pour i variant de 1 à n faire v prend la valeur 1 Afficher v v prend la valeur $\frac{9}{6-v}$ Fin pour **Fin algorithme**

Algorithme N° 3
Variables : v est un réel i et n sont des entiers naturels **Début de l'algorithme :** Lire n v prend la valeur 1 Pour i variant de 1 à n faire Afficher v v prend la valeur $\frac{9}{6-v}$ Fin pour Afficher v **Fin algorithme**

2. Traduire l'algorithme qui convient en langage Python.

Exercise 5

La durée d'une année (temps nécessaire à la Terre pour accomplir un tour complet autour du soleil) n'est pas un multiple entier du jour (temps nécessaire à la Terre pour faire un tour sur elle-même). Ainsi, un calendrier basé sur 365 jours par an se décalerait petit à petit et, au bout d'un certain temps, on finirait par avoir l'hiver en juillet et l'été en janvier... C'est pour cela que l'on introduit des années bissextiles : on rajoute exceptionnellement une journée à certaines années de manière à ce que la durée moyenne de l'année du calendrier se rapproche le plus possible de la durée nécessaire pour accomplir un tour autour du soleil.

La règle pour savoir si on ajoute un jour ou non (c'est-à-dire pour savoir si une année est bissextile ou pas) est la suivante : si le nombre entier associé à l'année (par exemple 2014) est divisible par 4 alors l'année est bissextile, sauf les multiples de 100 qui ne sont pas aussi multiples de 400.

Exemples : 2014 n'est pas bissextile (pas multiple de 4), 1900 n'est pas bissextile (divisible par 4 mais aussi par 100 et pas par 400) et 2000 est bissextile (divisible par 400).

1. Proposer une suite de tests logiques simples pour déterminer si une année est bissextile ou non. On ne demande pas encore d'écrire un code Python et on pourra présenter le résultat sous la forme d'un arbre.
2. Écrire une suite d'instructions permettant de créer une variable `annee` à laquelle on affectera la valeur 2015 puis une suite de tests logiques permettant d'afficher « Cette année est bissextile. » ou « Cette année n'est pas bissextile. » selon le cas. Un autre utilisateur doit pouvoir donner une valeur différente à la variable `annee`.
3. Écrire une fonction nommée `bissextile` qui prendra un paramètre `annee` en entrée et qui renverra `True` si l'année est bissextile et `False` si elle ne l'est pas. Documenter cette fonction (commentaires et aide).
4. En utilisant la fonction `bissextile`, écrire une suite d'instructions permettant de compter le nombre d'années bissextiles entre l'an 10 inclus et l'an 2015 exclu.
5. Définir une fonction à deux paramètres, `debut` et `fin`, qui compte le nombre d'années bissextiles entre une année initiale incluse (`debut`) et une année finale exclue (`fin`) et qui renvoie ce nombre. On supposera que la fonction `bissextile` est disponible.
6. En se servant de la fonction définie à la question 5, écrire une suite d'instructions permettant d'afficher à partir de quelle année il y aura eu 200 années bissextiles de plus que maintenant.

Exercise 6

Le code doit être écrit en langage Python. En particulier, on prendra soin de bien respecter l'indentation.

1. Écrire une fonction `factorielle` qui prend en argument un entier naturel n et renvoie $n!$ (on n'acceptera pas bien sûr de réponse utilisant la propre fonction factorielle du module `math` de Python).
2. Écrire une fonction `seuil` qui prend en argument un entier M et renvoie le plus petit entier naturel n tel que $n! > M$.
3. Écrire une fonction booléenne nommée `est_divisible`, qui prend en argument un entier naturel n et renvoie `True` si $n!$ est divisible par $n+1$ et `False` sinon.
4. On considère la fonction suivante nommée `mystere` :

```
def mystere(n) :
    s = 0
    for k in range(1,n+1) :
        s = s + factorielle(k)
    return s
```

   (a) Quelle valeur renvoie `mystere(4)` ?

   (b) Déterminer le nombre de multiplications qu'effectue `mystere(n)`.

   (c) Proposer une amélioration du script de la fonction `mystere` afin d'obtenir une complexité linéaire.

Exercise 7

On place une ampoule dans laquelle on a fait le vide dans une région de l'espace où règne un champ magnétique uniforme et stationnaire $\overrightarrow{B}$ (obtenu à l'aide de bobines de Helmholtz) et un champ électrique uniforme et stationnaire $\overrightarrow{E}$ perpendiculaire à $\overrightarrow{B}$ (obtenu grâce à deux électrodes portées à des potentiels différents). Un électron de charge $-e$ et de masse m est émis dans l'ampoule avec une vitesse initiale nulle. On choisit la position initiale de l'électron comme origine du repère cartésien dans lequel on travaille. La position de l'électron est repérée par ses coordonnées cartésiennes x, y et z dans ce repère et on pose

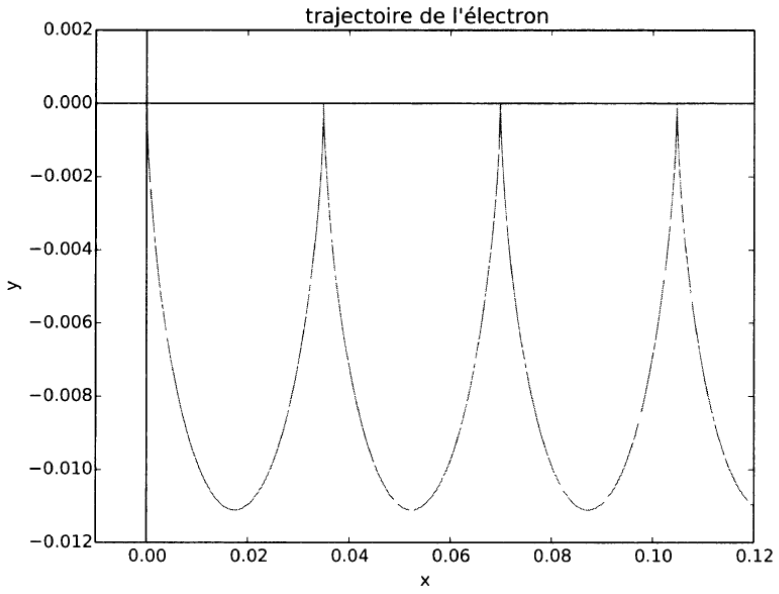
$$\overrightarrow{E} = E\,\vec{e}_y, \qquad \overrightarrow{B} = B\,\overrightarrow{e_z} \qquad \text{et} \qquad \omega = \frac{eB}{m}$$

L'application du principe fondamental de la dynamique à l'électron donne l'expression des coordonnées de l'électron en fonction du temps :

$$\begin{cases} x(t) &= \frac{E}{B}t - \frac{E}{B\omega}\sin(\omega t) \\ y(t) &= \frac{E}{B\omega}(\cos(\omega t) - 1) \\ z(t) &= 0 \end{cases}$$

Données : $E = 1{,}0\,\text{kV.m}^{-1}$, $B = 1{,}0\,\text{mT}$, $e = 1{,}6.10^{-19}\,\text{C}$, $m = 9{,}1.10^{-31}\,\text{kg}$

1. Donner la valeur de ω. Quel temps caractéristique peut-on associer à ω ? Donner sa valeur.
2. Écrire un programme en Python permettant d'obtenir la trajectoire de l'électron donnée sur la figure ci-dessous. Le programme devra renvoyer exactement la même figure, c'est-à-dire qu'elle doit avoir le même titre, les mêmes légendes sur les axes et les mêmes échelles. La trajectoire devra être représentée en rouge et les axes en noir.



Exercise 8

1. Écrire une suite de commandes qui permet d'échanger la première et la deuxième ligne de la matrice

$$A = \begin{pmatrix} 0 & 1 & 2 & 3 \\ -1 & 5 & -8 & 9 \\ -4 & 6 & -12 & 3 \\ -2 & 5 & -7 & -9 \end{pmatrix}$$

2. Écrire une suite de commandes qui permet de donner le plus grand élément de la deuxième ligne et le plus grand élément en valeur absolue de la troisième colonne de la matrice A.