Épreuves orales Math-info Centrale

Conseils : Tester toutes les fonctions au fur et à mesure qu'elles sont définies. Dans le cas où le programme tourne mais ne renvoie pas la valeur souhaitée, ajouter des "print(...)" au sein du programme pour déterminer l'origine du problème.

1) Tracé d'une fonction et d'une suite

a) Représentation de $t \longmapsto f(t)$ sur $[a, b]$

On suppose ici que f est une fonction déjà définie :

```
import numpy as np
def f(x) :  ...
a = ... ; b = ...
X = np.array(a,b+0.01,0.01)
### variante : n = int((b-a)/0.01) ; X = [ a+k*0.01 for k in range(n+1)]
Y = [f(x) for x in X]
import matplotlib.pyplot as plt
plt.plot(X,Y) ; plt.show()
```

b) Représentation de $(u_n)_{a \leq n \leq b}$

On suppose ici que $(u_n)_{a \leq n \leq b}$ est une fonction déjà définie :

```
def u(n) :  ...
N = [k for k in range(a,b+1)]   ### variante : N = list(range(a,b+1))
L = [u(k) for k in N]
import matplotlib.pyplot as plt
plt.plot(N,L) ; plt.show()
```

2) Intégrales paramétrées (seront sans doute déjà programmées)

```
import scipy.integrate as integr

def gamma(x) :
    def f(t) :
        return t**(x-1)*exp(-t)
    return integr.quad(f,0,inf)[0]
```

On peut ensuite représenter la fonction $x \longmapsto \Gamma(x) = \int_0^{+\infty} t^{x-1}e^{-t}\ dt$ sur un segment $[a, b]$.

3) Représentation d'une loi d'une v.a. X à valeurs dans $[\![0, m]\!]$ (ou par extension à valeurs dans $\mathbb{N}$)

a) On crée (donné en pratique) une fonction simulation `simul(m)` qui renvoie une valeur de X dans $[\![0, m]\!]$.
On considère N grand, par exemple $N = 1000$, et construit un tableau d'occurrences tab de sorte que `loi[j]` renvoie la proportion des valeurs obtenues valant j.

La loi de X est donnée par la représentation graphique du tableau loi :

```
    N = 1000 ; tab = [0]*(m+1)
    for k in range(N) : tab[simul(m)] += 1
    for j in range(m+1) :  tab[j] = loi[j]/N
    return tab
m = 10
V = [j for j in range(m+1)]
tab = loi(m)
W = [tab[j] for j in range(m+1)]
plt.plot(V,W) ; plt.show()
```

b) Estimation de l'espérance et de la variance**:**

```
def esperance(m) :
    N = 1000 ; s = 0 ; t = 0
    for k in range(N) :
        j = simul(m) ; s = s + j ; t = t + j**2
    esp = s/N ; var = t/N - esp**2
    return esp,var:
```

c) *Remarque* : Dans certains cas, on sait calculer le polynôme générateur.

Pour représenter la loi, il suffit alors d'afficher la liste des coefficients du polynôme générateur.

Exemple : Loi trinomiale de polynôme générateur $G_X(t) = \left(\frac{1+t+t^2}{3}\right)^n$ à valeurs dans $[\![0, 2n]\!]$.

```
from numpy.polynomial import *
def generateur(n) :
    P = Polynomial([1,1,1])/3
    return P**n
n = 10 ; G = generateur(n)
V = [j for j in range(2*n+1)]
W = G.coef     # renvoie la liste des coefficients du polynôme générateur G
plt.plot(V,W) ; plt.show()
```