Python td 12 - Révisions

Tous les programmes ci-dessous commenceront par :

```
import numpy as np
import numpy.random as rd
```

Exercice 1
Ecricome 2024
Soit $n \in \mathbb{N}^*$. On considère $X = (x_0, ..., x_n) \in \mathbb{R}^{n+1}$. Ecrire une fonction nommée `prodX`, prenant en entrée X, un entier i et un entier k (on suppose $i \leq k \leq n$) et qui renvoie le produit $\prod_{\substack{j=0 \\ j \neq i}}^{k} (x_i - x_j)$.

Exercice 2
EML 2023
Soit pour tout $n \in \mathbb{N}^*$, $S_n = \sum_{k=1}^{n} \frac{1}{k}$. On a montré (de façon classique à savoir dérouler quasi par coeur) que $S_n \sim_{n \to +\infty} \ln(n)$.

1. On considère la fonction Python suivante :

```
def rang(a):
    k=1
    s=1
    while s<a:
        k=k+1
        s=s+1/k
    return k
```

   Expliquer ce que produit l'appel rang(50).

2. Le code suivant

```
np.exp(49)
```

   renvoie : `1.90734657e+21`
   Expliquer rapidement ce que cela laisse penser si l'on fait l'appel `rang(50)`.

Exercice 3
Edhec 2015 ECS

On considère deux suites réelles $(I_n)_{n \in \mathbb{N}}$ et $(J_n)_{n \in \mathbb{N}}$ vérifiant :

$$I_n + I_{n+1} = \frac{1}{n+2}, \quad J_n + J_{n+1} = I_{n+1}, \quad J_0 = \frac{1}{2}, \quad I_1 = \ln(2)$$

Compléter le programme Python ci-dessous afin qu'il calcule les valeurs de I_n et J_n.

```
n=int(input("Entrer n supérieur ou égal à 2 :"))
I=np.log(2)
J=1/2
J=.....
for k in range(2,n+1):
    I=......
    J=.......
print(I,J)
```


Exercice 4
Ecricome 2015 ECS

On définit pour tout entier naturel n, les suites $(a_n)_{n \in \mathbb{N}}$ et $(b_n)_{n \in \mathbb{N}}$ par : $a_0 = \sqrt{\frac{3}{2}}$, $b_0 = \frac{1}{2}$ et

$$a_{n+1} = \sqrt{\frac{1 - b_n}{2}} \ (*) \quad \text{et} \quad b_{n+1} = \sqrt{\frac{1 + b_n}{2}} \ (**)$$

On admet que pour tout $n \in \mathbb{N}$,

$$9 \times 2^n . \frac{a_n}{2 + b_n} < \pi < 2^n . (2a_n + \frac{a_n}{b_n})$$

On admet enfin que $a_n \sim_{n \to +\infty} \frac{\pi}{3.2^n}$ et $b_n \sim_{n \to +\infty} 1$.

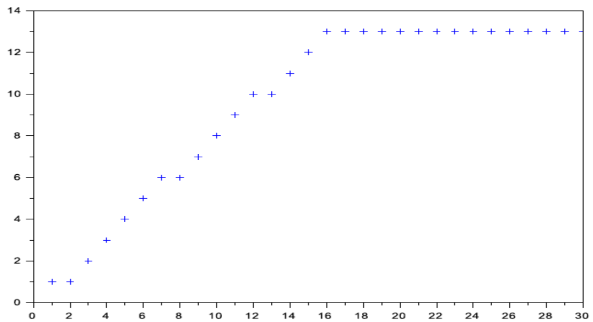
Justifier que les deux termes de l'encadrement précédent tendent vers π quand n tend vers $+\infty$.

Compléter la fonction Python suivante afin qu'elle retourne, à l'aide des relations $(*)$ et $(**)$ et des résultats admis, une approximation de π à e près, ainsi que le nombre d'itérations nécessaire.

```
def h(e):
    k=0
    a=\sqrt{3}{2}
    b=1/2
    while .......:
        a=......
        b=......
        k=.......
    x=......
    return x,k
```

On souhaite étudier l'évolution du nombre d'itérations nécessaires en fonction de la précision souhaitée. Ecrire une fonction Python qui prend comme paramètre d'entrée un paramètre p et qui retourne un vecteur de taille p qui contient les nombres d'itérations nécessaires pour les précisions 10^{-k}, pour $k \in \{1, 2, \cdots, p\}$.

On utilise la fonction précédente avec $p = 30$ et on représente graphiquement les valeurs obtenues. On obtient le graphe suivant :



Commenter ce graphe.

Exercice 5
Edhec 2015 ECE
Un joueur réalise une suite de lancers indépendants d'une pièce. Cette pièce donne Pile avec la probabilité p $(0 < p < 1)$ et Face avec la probabilité $q = 1 - p$.
On note N la variable aléatoire égale au rang d'apparition du premier Pile.
Si N prend la valeur n, le joueur place n boules numérotées de 1 à n dans une urne, puis il extrait une boule au hasard de cette urne. On dit que le joueur a gagné si le numéro porté par la boule tirée est impair et on note A l'événement "le joueur gagne". On appelle X la variable aléatoire égale au numéro porté par la boule extraite de l'urne.

1. Si m est un entier naturel, que renvoie la commande $2 * np.floor(m/2)$?

2. Compléter les commandes Python suivantes pour qu'elles simulent N et X et qu'elles renvoient l'un des deux messages "le joueur a gagné" ou "le joueur a perdu".

```
p=float(input(p"="))
N=......
X=........
if .......
   then print(".......")
   else print("........")
```

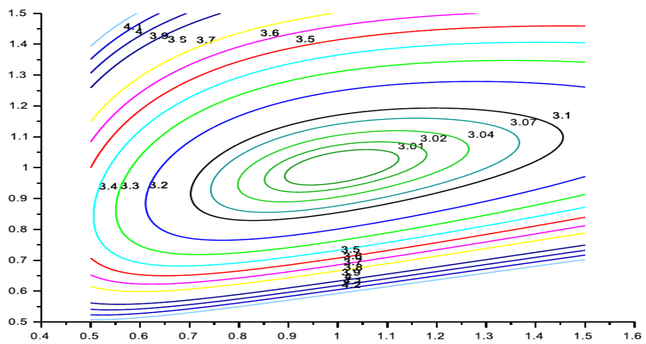
Exercice 6
Edhec 2024
On considère une suite $(J_n)_{n\in\mathbb{N}^*}$ vérifiant : $J_1 = \frac{\pi}{2}$ et pour tout $n \in \mathbb{N}^*$, $J_n = 2n(J_n - J_{n+1})$. Compléter le programme suivant pour qu'il renvoie la valeur de J_n :

```
def suiteJ(n):
   J=-----
   for k in range(2,n+1):
      J=--------
   return J
```

Exercice 7
Ecricome 2019 ECE
On considère la fonction f définie sur l'ouvert $]0;+\infty[\times]0;+\infty[$ par :

$$\forall (x,y) \in (\mathbb{R}_+^*)^2, \quad f(x,y) = \frac{x}{y^2} + y^2 + \frac{1}{x}$$

1. On utilise Python pour tracer les lignes de niveau de la fonction f. On obtient le graphe suivant :



   Etablir une conjecture à partir du graphique quant à l'existence d'un extremum local pour f, dont on donnera la nature, la valeur approximative et les coordonnées du point où il semble être atteint.

2. Pour tout entier non nul, on note h_n la fonction définie sur $\mathbb{R}_+^*$ par :

$$\forall x > 0, \quad h_n(x) = f(x^n, 1) = x^n + 1 + \frac{1}{x^n}$$

   On admet que l'équation $h_n(x) = 4$ admet exactement deux solutions, notées u_n et v_n et vérifiant : $0 < u_n < 1 < v_n$.

   (a) Ecrire une fonction Python d'en-tête `def h(n,x):` qui renvoie la valeur de $h_n(x)$.

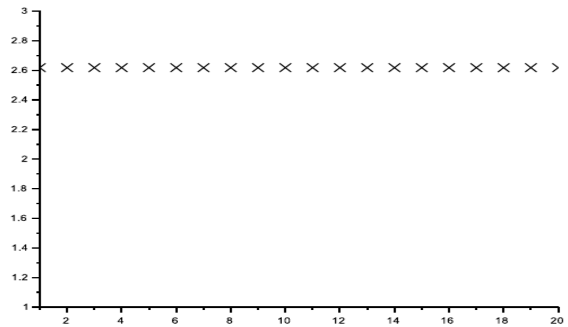
   (b) Compléter la fonction suivante, pour qu'elle renvoie une valeur approchée à 10^{-5} près de v_n par la méthode de dichotomie lorsqu'on lui fournit $n \geq 1$ en entrée :


```
def v(n):
   a=1
   b=3
   while (b-a)>10**(-5):
        c=(a+b)/2
        if h(n,c)<4 :
            ........
        else :
            .........
return ......
```

   (c) A la suite de la fonction v, on écrit le code suivant :

```
X=np.arange(1,21,1)
Y=np.zeros(1,20)
for k in range(1,21):
    Y[k]=v[k]**k
plt.plot(X,Y)
```

   A l'exécution du programme, on obtient la sortie graphique suivante :



   Expliquer ce qui est affiché sur le graphique ci-dessus. Que peut-on conjecturer ?

Exercice 8
HEC 2017 ECS

On rappelle que si pour tout $n \in \mathbb{N}^*$, $Z_n \hookrightarrow \mathcal{B}(n,p)$, en notant $\overline{Z_n} = \frac{Z_n}{n}$, la suite $(\overline{Z_n})_{n\geq 1}$ converge en probabilités vers p. Si f est une fonction continue, on a alors $(f(\overline{Z_n}))_{n\geq 1}$ qui converge vers $f(p)$. On considère le programme suivant :

```
def f(x):
   if x==0 :
      y=0
   else :
      y=-x*np.log(x)
   return y
p=0.4
n=100;   N=1000;   S=0
for k in range(1,N+1):
    S=S+f(rd.binomial(n,p)/n)
print(S/N)
```

Ce programme affiche la valeur approchée d'une certaine quantité, laquelle ?
Cette valeur approchée est le résultat de la mise en oeuvre de certaines méthodes, lesquelles ?