TP1

Rappels de 1ère année : suites et sommes

Travail sur les listes simples

Génération de listes

```
L=[k+1 for k in range(10)]
X=[np.log(k) for k in L]
Y=[k**2 for k in L]
Z=[k**2/np.log(k) for k in L]
```

Contenus des variables X et Y :
L=[1,2,3,4,5,6,7,8,9,10],

X=[$\ln 1, \ln 2, \ln 3, \ln 4, \ln 5, \ln 6, \ln 7, \ln 8, \ln 9, \ln 10$],

Y=[$1^2, 2^2, 3^2, 4^2, 5^2, 6^2, 7^2, 8^2, 9^2, 10^2$]

Représenter graphiquement des listes dans un même graphique

Reprenons les listes contenues dans les variables X et Y.

```
import matplotlib.pyplot as plt
    plt.grid()
    plt.plot(X,'+')
    plt.plot(Y,'.')
    plt.plot(Z,'.')
    plt.show()
```

Exercice 1.1 (Croissances comparées).

1. Que contient la variable Z ?
2. Quelle croissance comparée le script précédent permet-il d'illustrer ?
3. Mettre en évidence deux autres croissances comparées.

Suites récurrentes

Prenons l'exemple de la suite définie par $u_0 = 2$ et $\forall n \in \mathbb{N}, \quad u_{n+1} = u_n - \left(\ln(u_n)\right)^2$.

Génération d'un seul terme

```python
import numpy as np
def suite_u(n):
    u=2
    for k in range (n):
        u=u-np.log(u)**2
    return u
```

Fonction prenant en argument un entier n et qui renvoie le terme u_n.

Génération d'une liste de termes consécutifs

```python
def liste_u(n):
    L=[]
    u=2
    for k in range(n):
        L.append(u)
        u=u-np.log(u)**2
    return L
```

Fonction prenant en argument un entier n et qui renvoie la liste des $n+1$ premiers termes de la suite $(u_n)_{n\in\mathbb{N}}$.

Représentation graphique

```python
import matplotlib.pyplot as plt
def graphe_u(n):
    L=liste_u(n)
    plt.grid()
    plt.plot(L,'+')
    plt.show()
```

Exercice 1.2.
On considère la suite définie par: $u_0 = 10$, et $\forall\, n \in \mathbb{N}, \quad u_{n+1} = \frac{1}{2}\left(u_n + \frac{2}{u_n}\right)$.

1. Écrire une fonction Python, d'en-tête `def suite_u(n):`, qui prend en argument un entier n et renvoie le n^{ieme} terme de cette suite. Calculer à l'aide de Python u_5 et u_10.

2. Écrire une fonction Python, d'en-tête `def liste_u(n):`, qui prend en argument un entier n et renvoie les $n+1$ premiers termes de la suite $(u_n)_{n\in\mathbb{N}}$. Afficher la liste des 15 premiers termes.

3. Écrire une fonction Python, d'en-tête `def graphe_u(n):`, qui prend en argument un entier n et renvoie le graphique des $n+1$ premiers termes de la suite $(u_n)_{n\in\mathbb{N}}$.

4. Que pouvez-vous conjecturer sur la suite $(u_n)_{n\in\mathbb{N}}$?

Suites vérifiant une relation de récurrence linéaire d'ordre 2

La célèbrissime suite de Fibonacci est définie par:

$$F_0 = F_1 = 1, \quad \forall\, n \in \mathbb{N}, \quad F_{n+2} = F_{n+1} + F_n.$$

```python
def fibo(n):
    u,v=1,1
    for k in range(n):
        u,v=v,u+v
    return u
```

Fonction prenant en argument un entier n et qui renvoie le terme F_n.

EXERCICE 1.3 (Sur la suite de Fibonacci)**.**

1. Calculer avec `Python` les termes F_8, F_{15} et F_{30} Que pouvez-vous conjecturer sur la nature de la suite $(F_n)_{n\in\mathbb{N}}$?
2. Écrire une fonction `Python`, d'en-tête `def ratio_fibo(n):` qui prend en argument un entier n et renvoie la valeur de $\dfrac{F_{n+1}}{F_n}$. Exécuter votre fonction avec $n = 10, 20, 50, 100$.
3. Calculer avec `Python` une valeur approchée pour $\frac{\sqrt{5}+1}{2}$.
4. Que conjecturez-vous concernant la limite $\lim\limits_{n\longrightarrow+\infty} \dfrac{F_{n+1}}{F_n}$?

Suites des sommes partielles de séries

```python
def somme(n):
    S=0
    for k in range (n):
        S=S+1/2**k
    return S
```

Fonction prenant en argument un entier n et qui renvoie la somme $\sum\limits_{k=0}^{n} \dfrac{1}{2^k}$.

EXERCICE 1.4 (Séries de Riemann)**.**
Pour tout $\alpha > 0$ et tout $n \in \mathbb{N}^*$ on pose :

$$S_n(\alpha) = \sum_{k=1}^{n} \frac{1}{k^\alpha}.$$

1. Écrire une fonction `Python`, d'en-tête `def riemann(n,alpha):`, qui prend en argument un entier n et un réel $\alpha > 0$ et renvoie la valeur de $S_n(\alpha)$.
2. Écrire une fonction `Python`, d'en-tête `def riemann_liste(n,alpha):` qui prend en argument un entier n et un réel α et renvoie la liste des n premiers termes de la suite valeur de $\big(S_n(\alpha)\big)_{n\in\mathbb{N}}$.
3. Représenter graphiquement les 100 premiers termes $\big(S_n(\alpha)\big)_{n\in\mathbb{N}}$ pour différentes valeurs de α.
4. **(Plus dur)** Écrire un script permettant de représenter dans un seul graphique trois graphiques pour des valeurs de α bien choisies.

Algorithmes de seuil

On considère ici la suite définie par : $u_0 = 2$ et $\forall n \in \mathbb{N}, \quad u_{n+1} = u_n + \dfrac{1}{u_n}$.

```python
def seuil(A):
    n=0
    u=2
    while u<A:
        u=u+1/u
        n=n+1
    return n
```

Fonction prenant en argument un réel $A > 0$ et qui renvoie le premier entier n tel que $u_n \geqslant A$.

EXERCICE 1.5.
On considère la suite réelle $(u_n)_{n\in\mathbb{N}}$ définie par $u_0 = 2$ et, pour tout $n \in \mathbb{N}$,

$$u_{n+1} = (u_n + \ln(u_n))e^{u_n - 1}$$

On admet que la suite diverge vers $+\infty$.

Écrire une fonction Python qui calcule et affiche le plus petit entier naturel n tel que $u_n \geqslant 10^5$.

EXERCICE 1.6. *Adapté d'ECRICOME 2006*
On considère la fonction f définie pour tout réel x par : $f(x) = x + 1 + 2e^x$.
On s'intéresse à la suite $(u_n)_{n\in\mathbb{N}}$ définie par son premier terme $u_0 = -1$ et par la relation

$$\forall n \in \mathbb{N} \quad u_{n+1} = u_n - \frac{f(u_n)}{f'(u_n)}$$

(...Après étude mathématiques...), on montre que l'équation $f(x) = 0$ admet une unique solution $\alpha \in \mathbb{R}$ et

$$\forall n \in \mathbb{N}, \quad 0 \leqslant u_n - \alpha \leqslant \frac{1}{e^{2^n - 1}}$$

Écrire une fonction Python permettant, lorsque l'entier naturel p est donné par l'utilisateur, de calculer une valeur approchée de α, de telle sorte que l'on ait : $0 \leqslant u_n - \alpha \leqslant 10^{-p}$.

Suites adjacentes

Le cas des suites adjacentes est un peu particulier. En effet, lorsqu'on a montré que deux suites $(u_n)_{n\in\mathbb{N}}$ et $(v_n)_{n\in\mathbb{N}}$ sont adjacentes, on sait d'une part qu'elles convergent toutes deux vers une même limite ℓ et d'autre part que :

$$\forall n \in \mathbb{N}, \quad u_n \leqslant l \leqslant v_n.$$

Ainsi, pour trouver une valeur approchée de la limite commune ℓ à 10^{-3} près, il suffit de connaître la valeur de u_n (ou de v_n) quand $|v_n - u_n| \leqslant 10^{-3}$.
On adaptera donc le script avec la boucle `while` avec une condition du type : `abs(v-u) >= eps`

EXERCICE 1.7. (Plus dur) *Adapté d'ECRICOME 2007*
Soit $(H_n)_{n\in\mathbb{N}^*}$ la suite des sommes partielles de la série harmonique : $H_n = \displaystyle\sum_{k=1}^{n} \frac{1}{k}$.
On considère les suites $(u_n)_{n\in\mathbb{N}^*}$ et $(v_n)_{n\in\mathbb{N}^*}$ définies par :

$$u_n = H_n - \ln(n) \quad \text{et} \quad v_n = u_n - \frac{1}{n}.$$

(...Après étude mathématiques...), on montre que (u_n) et (v_n) sont adjacentes et convergent vers un réel γ, appelé constante d'Euler.

1. Écrire un programme en langage Python permettant d'afficher le terme u_n lorsque l'on rentre l'entier n.
2. En utilisant ce qui précède, écrire un programme en langage Python permettant d'afficher une valeur approchée à 10^{-4} de γ.