T.P. IV - Suites. . .

Code Capytale : `7db3-1087927`

I - Ce qu'il faut savoir

* **Définir** une suite par :
  * son terme général,
    la structure `for i in range(a, b):` permet de faire parcourir à `i` les valeurs de `a` à `b`-1.
  * récurrence à l'aide de la définition d'une **fonction**,
    La fonction peut être définie de manière externe avec le mot-clé `def` ou alors être définie à chaque itération.
  * récurrence pour des suites **imbriquées**,
  * récurrence linéaire avec des **matrices**.
* **Tracer** les termes successifs et **interpréter** un comportement asymptotique.
* Déterminer un **seuil**
  Les boucles conditionnelles `while` permettent d'interrompre le calcul dès qu'une condition est satisfaite.

II - . . . définies par leur terme général

Exercice 1. (Étude de suite) Pour tout n entier naturel, on pose $c_n = 2 - \frac{3^n+25}{4^{n-1}}$.

1. Compléter le code suivant pour qu'il affiche le graphe des points de coordonnées $((n, c_n))_{5\leqslant n\leqslant 21}$.

```
import matplotlib.pyplot as plt

def c(n):
    return ...

X = range(..., ...)
Y = [c(n) for n in X]

plt.figure()
plt.plot(..., ..., '.')
plt.show()
```

2. Déterminer $\lim_{n\to+\infty} c_n$.

3. On considère le code Python suivant :

```
n = 1
c = 2 - (3**n + 25)/4**(n-1)
while c < 1.95 :
    n = n + 1
    c = 2 - (3**n + 25)/4**(n-1)
print(n)
```

On obtient l'affichage suivant : 16.
Interpréter le résultat dans le contexte de l'énoncé.

Exercice 2. (Terme général, Seuil) On considère la suite $(c_n)_{n\in\mathbb{N}^*}$ définie par

$$\forall\, n \in \mathbb{N}^*,\ c_n = 1 - \frac{2^n - 1}{3^{n-1}}.$$

1. Compléter le code suivant pour qu'il affiche le graphe des points de coordonnées $((n, c_n))_{5\leqslant n\leqslant 21}$.

```
import matplotlib.pyplot as plt

def c(n):
    return ...

X = range(..., ...)
Y = [... for n in X]

plt.figure()
plt.plot(..., ..., '.')
plt.show()
```

2. Évaluer la suite d'instructions suivante puis interpréter le résultat.

```
n = 1
c = 1 - (2**n - 1) / 3**(n-1)
while c < 0.95:
    n = n + 1
    c = 1 - (2**n - 1)/3**(n-1)

print(n)
```

Exercice 3. (Terme général, Seuil) Soit (a_n) et (b_n) les suites définies pour tout n entier naturel par

$$\begin{cases} a_n &= \frac{1}{3}\left(1+\frac{2}{4^n}\right) \\ b_n &= \frac{1}{3}\left(1-\frac{2}{4^n}\right) \end{cases}$$

1. Compléter le script ci-dessous pour qu'il affiche sur une même figure les graphes des points de coordonnées $((n, a_n))_{1\leqslant n\leqslant 20}$ et $((n, b_n))_{1\leqslant n\leqslant 20}$.

```
import matplotlib.pyplot as plt

def a(n):
    return ...

def b(n):
    return ...

X = range(..., ...)
Ya = [a(n) for n in X]
Yb = [b(n) for n in X]

plt.figure()
plt.plot(X, Ya, '.')
plt.plot(X, Yb, 'd')
plt.show()
```

2. Compléter le script ci-dessous afin qu'il calcule le plus petit entier naturel n tel que l'on ait à la fois : $a_n \leqslant 0{,}334$ et $b_n \geqslant 0{,}333$.

```
n = 0
a = 1
b = ...
while ...:
    n = ...
    a = 1/3 * (1 + 2/4**n)
    b = ...

print(...)
```

III - . . . récurrentes simples

Exercice 4. (Suite arithmético-géométrique) Soit (u_n) la suite définie par $u_0 = 1$ et

$$\forall\, n \in \mathbb{N},\, u_{n+1} = \frac{1}{2}u_n + 3.$$

1. Compléter le code suivant qui permet de calculer le n-ième terme de la suite puis d'afficher les valeurs de u_n pour $0 \leqslant n \leqslant 20$. Quelle conjecture pouvez vous effectuer sur le comportement de la suite ?

```
import numpy as np

def u(n):
    c = 1
    for i in range(1, ...):
        c = 1 / 2 * c + ...
    return c

X = np.arange(0, ..., 1)
Y = [u(n) for n in X]

plt.figure()
plt.plot(..., ..., 'o')
plt.show()
```

2. On souhaite déterminer puis afficher le plus petit entier naturel n_0 tel que $u_{n_0} \geqslant 5{,}5$. Compléter le code suivant :

```
n = 0
c = 1
while c < ...:
    n = ...
```

```
    c = ...

print(...)
```

Exercice 5. (Série géométrique) Soit $q \in \mathbb{R}$. On souhaite étudier les suites définies par $v_0 = 1$ et, pour tout n entier naturel, $v_{n+1} = qv_n$.

1. a) Compléter le code suivant qui permet définir une fonction `suite_geom` telle que l'appel `suite_geom(n, q)` renvoie le terme v_n :

```
def suite_geom(n, q):
    v = 1
    for i in range(..., ...):
        v = ...
    return ...
```

b) Compléter le code qui permet d'afficher les valeurs de v_n lorsque :
* $n = 10$ et $q = 0,1$.
* $n = 100$ et $q = 2$.
* $n = 110$ et $q = 0,5$.

```
print("n = 10, q = 0.1", suite_geom(..., ...))
print("n = 100, q = 2", suite_geom(..., ...))
print("n = 110, q = 0.5", suite_geom(..., ...))
```

2. On souhaite maintenant cacluler les termes successifs de la suite $(w_n)_{n\in\mathbb{N}}$ définie par $w_0 = 0$ et $w_n = \sum_{k=1}^{n} v_k$ pour tout entier naturel n.

a) Compléter le code de la fonction suivante qui renvoie le terme w_n :

```
def serie_geom(n, q):
    s = ...
    for i in range(..., ...):
        s = s + ...
    return s
```

b) On souhaite représenter graphiquement les termes $w_0, \dots, w_{100}$ de cette suite pour $q = 0.1$. Compléter le code suivant :

```
import matplotlib.pyplot as plt
n = ...
s = [serie_geom(...) for n in range(..., ...)]

plt.plot(range(...), ...)
...
```

c) Reprendre la question précédente avec $q = 1.2$.

IV - ... récurrentes du type $u_{n+1} = f(u_n)$

Exercice 6. On définit la suite (u_n) par $u_0 = 1$ et pour tout n entier naturel, $u_{n+1} = \ln(1 + u_n^2)$.

1. Compléter la suite d'instructions suivante pour qu'elle calcule et affiche la valeur de u_5.

```
import numpy as np
n = ...
u = ...
for k in range(1, n+1):
    u = ...

print(u)
```

2. On admet que la suite (u_n) converge vers 0. Exécuter la suite d'instructions suivante.

```
import numpy as np
n = 0
u = 1
while u >= 0.0001:
    u = np.log(1 + u**2)
    n = n + 1

print(n)
```

Le résultat affiché est 6. Quelle est la signification de ce résultat ?

Exercice 7. On considère la fonction g définie sur $[1, 2]$ par $g(x) = \ln(x + 2)$ et la suite (u_n) définie par $u_0 = 1$ et $u_{n+1} = g(u_n)$ pour tout $n \in \mathbb{N}$.

1. Compléter la suite d'instructions suivante pour qu'elle stocke les valeurs de u_0 à u_{20} et qu'elle les représente graphiquement.

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

# Initialise une liste ne contenant que des 1
U = np.ones((21, 1))
for i in range(1, 21):
    U[i] = ...

X = np.arange(0, 21, 1)
plt.figure()
plt.plot(..., ..., '+')
plt.show()
```

2. Que conjecturer sur le sens de variation et sur la limite de la suite (u_n) ?

Exercice 8. Soit (u_n) la suite définie par $u_0 = 1$ et

$$\forall\, n \in \mathbb{N},\, u_{n+1} = \frac{u_n}{1 + u_n + u_n^2}.$$

On admettra que (u_n) est décroissante et converge vers 0.
Compléter la suite d'instructions suivante pour qu'elle affiche le plus petit entier naturel n non nul tel que $u_n \leqslant 1/1000$.

```python
def f(x):
    return x/(1 + x + x**2)

u = ...
n = ...
while u ...:
    u = ...
    n = ...

print(...)
```

V - ... récurrentes dépendant du rang

Exercice 9. On considère la suite (I_n) définie par $I_0 = \frac{1}{2} - \frac{1}{2\,\mathrm{e}}$ et

$$\forall\, k \geqslant 0,\, I_{k+1} = kI_k - \frac{1}{2\,\mathrm{e}}.$$

Compléter la suite d'instructions suivante pour qu'elle calcule et affiche la valeur de I_{21}.

```python
import numpy as np
n = ...
I = 1/2 - 1/(2 * np.exp(1))
for k in range(1, n+1):
    I = ...

print(I)
```

Exercice 10. Soit (I_n) la suite définie par $I_0 = 1$ et

$$\forall\, n \geqslant 0,\, I_{n+1} = \frac{1}{\mathrm{e}} + (n+1)I_n.$$

Écrire une suite d'instructions qui calcule et affiche la valeur de I_{10}.

VI - ... imbriquées

Exercice 11. On considère les suites (u_n) et (v_n) définies par $u_0 = 0$, $v_0 = 1$ et

$$\forall\, n \in \mathbb{N},\, \begin{cases} u_{n+1} &= \frac{u_n + v_n}{2} \\ v_{n+1} &= \frac{u_{n+1} + v_n}{2} \end{cases}$$

Compléter la suite d'instructions suivante qui permet de déterminer u_n et v_n pour $n = 50$.

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

print("u50", u)
print("v50", v)
```

Exercice 12. Soit $(u_n)_{n\geqslant 0}$ et $(v_n)_{n\geqslant 0}$ deux suites définies par $u_1 = 1$, $v_1 = 2$ et

$$\forall\, n \in \mathbb{N},\ u_{n+1} = \frac{u_n^2}{u_n + v_n} \text{ et } v_{n+1} = \frac{v_n^2}{u_n + v_n}.$$

1. Compléter la suite d'instructions suivantes afin qu'elle calcule et affiche les valeurs de u_{10} et v_{10}.

```
import numpy as np

n = ...
u = 1
v = 2
for k in range(2, n+1):
    a = u
    u = ...
    v = ...

print("u10", u)
print("v10", v)
```

2. On considère le programme précédent avec les instructions supplémentaires :

```
import numpy as np
import matplotlib.pyplot as plt
n = ...
u = 1
v = 2
s = np.zeros((n+1, 1))
s[1] = u
for k in range(2, n+1):
    a = u
    u = ...
    v = ...
    s[k] = u

X = np.arange(0, n+1)
# Calcule la somme cumulee de la matrice ligne s :
Y = np.cumsum(s)
plt.figure()
plt.plot(X, Y)
plt.show()
```

a) Que contiennent les variables `s` et `y` à l'issue du programme ?

b) Quel résultat le graphique obtenu permet-il de conjecturer ?

VII - . . . récurrentes doubles

Exercice 13. (Suite récurrente double) On considère la suite définie par $u_0 = 0$, $u_1 = 1$ et $\forall\, n \in \mathbb{N}^*$, $u_{n+1} = 4u_n + 2u_{n-1}$.
Complter les 3 lignes du script Python ci-dessous pour qu'il calcule et affiche la valeur de u_{10}.

```
v = 0
u = 1
for i in range(..., ...):
    a = u
    ...
    v = a

print(u)
```

VIII - . . . & fonctions : la dichotomie

Exercice 14. (Exemple de dichotomie) On pose $h(x) = x^3 + \frac{1}{x^3} - 3$.

1. Montrer que h est strictement croissante sur $]0, 1]$ et strictement décroissante sur $[1, +\infty[$.

2. Montrer que l'équation $h(x) = 0$ possède une unique solution α dans l'intervalle $[1, +\infty[$.

3. Compléter le code Python suivant pour qu'il renvoie une valeur approchée à 10^{-5} près de α par la méthode de dichotomie.

```
def h(x):
    return ....

a = ...
b = ...
```

```
while (b - a) ...:
    m = ...
    if h(m) * h(a) <= 0:
        b = ...
    else:
        ...

print(...)
```