Chapitre 2 : Les suites

1 Suites récurrentes d'ordre 1

1.1 Suites récurrentes et boucles

On considère une suite $(u_n)_{n \in \mathbb{N}}$ définie par : $\begin{cases} u_0 \in \mathbb{R} \\ \forall\, n \in \mathbb{N},\ u_{n+1} = f(u_n) \end{cases}$

On veut calculer les valeurs successives de (u_n), stockées dans la variable u :

Initialisation : **u = u0 ; n = 0**
Formule de récurrence : **u = f(u)**

Remarque : si la fonction f n'a pas été définie auparavant, il faut remplacer f(u) par son expression.

Exemple : Soit $(u_n)_{n \in \mathbb{N}}$ la suite définie par : $\begin{cases} u_0 = 1 \\ \forall\, n \in \mathbb{N},\ u_{n+1} = u_n + u_n^2 \end{cases}$

Déterminer le premier entier n tel que $u_n \geq 10^4$.

```
u=1;n=0
while u<10**4:
        n=n+1
        u=u+u**2
print(n)
```

(On trouve n=5)

Remarque :
Si on veut conserver les valeurs de la suite, on peut les stocker dans une liste, en utilisant append

Ex : suite de l'exemple précédent. Stocker dans un tableau les valeurs de $(u_0, u_1, \ldots, u_8)$.

```
u=1
suite=[1]
for i in range(1,9):
   u=u+u**2
   suite.append(u)
print(suite)
```

1.2 Suites et valeurs approchées

Rappel : Soit x et y deux réels et ε un réel strictement positif.

Alors **y est une valeur approchée de x à epsilon près** signifie que : $|\mathbf{y - x}| \leq \boldsymbol{\varepsilon}$.

Exemple :

Soit $(u_n)_{n \in \mathbb{N}}$ la suite définie par : $\begin{cases} u_0 = 2 \\ \forall\ n \in \mathbb{N},\ u_{n+1} = \frac{1}{2}\left(u_n + \frac{2}{u_n}\right) \end{cases}$

Déterminer le premier entier n tel que u_n est une valeur approchée de $\sqrt{2}$ à 10^{-3} près.

```
import numpy as np
n=0;u=2
while abs(u-np.sqrt(2))>10**(-3):
   n=n+1
   u=(u+2/u)/2
print(n)
```

⟶ on trouve n = 3

Remarque :
Il est parfois nécessaire de travailler par **condition suffisante** (et non nécessaire). Dans ce cas, on commence par écrire une phrase du type "Pour que … il suffit que …"

Ex :
Soit f la fonction définie sur]0; + ∞[par : f(x) = 2 + ln(x).
On admet que f admet un unique point fixe α sur [1; + ∞[
Soit u_n la suite définie par : $\begin{cases} u_0 = 3 \\ u_{n+1} = f(u_n), \forall\ n \in \mathbb{N} \end{cases}$
On admet que $\forall\ n \in \mathbb{N}, \left|u_n - \alpha\right| \leq \frac{1}{3^n}$.
Déterminer à l'aide de Python une valeur approchée de α à 10^{-3} près.

Remarques
_ u_n tend vers α : une valeur approchée de α sera donc donnée par u_n, pour un n assez grand.
_ α est inconnu, donc une instruction du type "while abs(u – alpha) " n'est pas possible !

Pour que $\left|u_n - \alpha\right| \leq 10^{-3}$, il suffit que $\frac{1}{3^n} \leq 10^{-3}$.

```
import numpy as np
n=0;u=3;
while 1/(3**n)>10**(-3):
        n=n+1
        u=2+np.log(u)
print(u)
```
⟶ On trouve $\alpha \approx 3{,}1461436\ldots$

Remarque : On peut également déterminer par avance le nombre d'itérations :
(Rappel : int(x) donne la partie entière du réel x)

$\frac{1}{3^n} \leq 10^{-3} \Leftrightarrow 3^n \geq 10^3 \Leftrightarrow n.\ln(3) \geq 3\ln(10) \Leftrightarrow n \geq \frac{3\ln(10)}{\ln(3)}$ (car ln(3) > 0)

```
from numpy import log
u=3;
n0=int(3*log(10)/log(3))+1
for i in range(n0):
        u=2+log(u)
print(u)
```

2. Suites récurrentes d'ordre 2

On considère une suite $(u_n)_{n \in \mathbb{N}}$ définie par : $\begin{cases} u_0 \in \mathbb{R} \\ u_1 \in \mathbb{R} \\ \forall\, n \in \mathbb{N},\ u_{n+2} = f(u_n, u_{n+1}) \end{cases}$

Pour calculer les valeurs de (u_n), il faut conserver deux valeurs successives, stockées dans u et v.

Initialisations : **u = u0 ; v = u1**
Relations de récurrence : **w=f(u, v)**
u =v
v =w

Explication :

	u	v	w
Avant le tour de boucle			
w = f(u, v)			
u = v			
v = w			

Remarque : v commençant à u_1, il faut n-1 itérations pour que v contienne u_n.

Exemple : Soit $(u_n)_{n \in \mathbb{N}}$ la suite définie par : $\begin{cases} u_0 = 2 \\ u_1 = -1 \\ \forall n \in \mathbb{N},\ u_{n+2} = u_{n+1} + 3u_n \end{cases}$. Déterminer u_{10}

```
u=2;v=-1
for i in range(2,11):
        w=v+3*u
        u=v
        v=w
print(v)
```
⟶ On trouve $u_{10} = 1889$

Remarque :
En Python, l'instruction : **a,b = c,d** stocke <u>en même temps</u> c dans a et d dans b.
Il est donc possible d'écrire : **u, v = v, f(u,v)**

Dans l'exemple précédent :