

PYTHON : LES BOUCLES FOR ET WHILE

EXERCICE 1 Tester les algorithmes suivants, puis les reprogrammer avec des boucles for.

```

1 #1
2 i=1
3 while i<10:
4     i+=1
5     print(i)
6 print('fini')
7
8 #2
9 i=1
10 while i<10:
11     print(i)
12     i+=1
13 print('fini')
```

```

1 #3
2 i=0
3 while i<=10:
4     print(i)
5     i+=1
6 print('fini')
7
8 #4
9 i=0
10 while i<11:
11     print(i)
12     i+=2
13 print('fini')
```

EXERCICE 2 (marche arrière)

- 1) Programmer une boucle **while** qui écrit les nombres de 100 à 1 dans l'ordre décroissant.
- 2) Faire de même avec une boucle **for**.

Exercice type à savoir refaire rapidement :

EXERCICE 3 (suite définie par une relation de récurrence)

Soit la suite définie par $u_0 = 1$ et $\forall n \in \mathbb{N}, u_{n+1} = \sqrt{u_n + n}$

- 1) (*mathématiques*) Justifier que la suite est bien définie pour tout $n \in \mathbb{N}$.
- 2) Écrire une fonction Python **suite(n)** qui prend en argument l'entier naturel n et qui renvoie la valeur de u_n .
On doit trouver $u_{100} \approx 10,45990\dots$
- 3) Écrire une fonction Python **somme(n)** qui renvoie la valeur $\sum_{k=0}^n u_k$.
On doit trouver $S_{25} \approx 94,7077\dots$
- 4) Écrire une fonction Python **croissante(n)** qui renvoie si la suite est croissante jusqu'au rang n .
- 5) (*mathématiques*)
 - (a) Justifier que la suite u est croissante.
 - (b) Montrer que la suite u diverge vers $+\infty$.
 - (c) En déduire que $\forall M \in \mathbb{R}, \exists n \in \mathbb{N}$, tel que $u_n > M$.
- 6) Écrire une fonction **rangSup(M)** qui prend en argument un nombre M et renvoie le premier indice n tel que $u_n \geq M$.
- 7) Écrire une fonction **rangInf(M)** qui prend en argument un nombre M et renvoie le dernier indice n tel que $u_n < M$.
- 8) Écrire une fonction **termeSup(M)** qui prend en argument un nombre M et qui renvoie le premier terme u_n de la suite tel que $u_n \geq M$.
- 9) Écrire une fonction **termeInf(M)** qui prend en argument un nombre M et qui renvoie le dernier terme u_n de la suite tel que $u_n < M$.

EXERCICE 4 (logarithme)

- 1) Écrire une fonction **log2** qui prend en argument un entier n , avec $n \geq 1$ et calcule le plus grand entier k tel que $2^k \leq n$.
- 2) Vérifier la fonction avec l'entier 128.
- 3) Vérifier la fonction avec l'entier 9876543210123456789.

EXERCICE 5 (Suite de Syracuse)

Une suite de Syracuse est définie par

$$u_0 \in \mathbb{N}^*, \quad \text{et} \quad \forall n \in \mathbb{N}^*, \quad u_{n+1} = \begin{cases} \frac{u_n}{2} & \text{si } u_n \text{ est pair} \\ 3u_n + 1 & \text{si } u_n \text{ est impair} \end{cases}$$

Dans la suite, on considérera que la conjecture de Syracuse est juste : pour toute condition initiale $u_0 \in \mathbb{N}^*$, la suite atteint la valeur 1.

- 1) Programmer une fonction **syracuse(u0, n)** qui prend en argument un entier $u_0 \geq 1$, et un entier $n \in \mathbb{N}$ et qui calcule le terme u_n de la suite de Syracuse.
- 2) Programmer une fonction **tempsVol(u0)** qui prend en argument un entier $u_0 \geq 1$, et qui renvoie le premier entier n tel que $u_n = 1$.
- 3) Programmer une fonction **maximum(u0)** qui prend en argument un entier $u_0 \geq 1$, et qui renvoie le maximum de la suite (u_n) .
- 4) Programmer une fonction **tpsVolMax(n)** qui prend en argument un entier $n \geq 1$, et qui renvoie le plus petit entier $p \in \llbracket 1, n \rrbracket$ pour lequel la suite de premier terme $u_0 = p$ a le plus grand temps de vol.

EXERCICE 6 (S'il reste du temps)

Nous chargeons le module qui permet de tracer des courbes en Python.

```
1 from matplotlib.pyplot import *
2
3 plot(3,5,'ro') # trace un rond (o) rouge (r) de coordonnées (3,5)
4 plot(1,-3,'b.') # trace un point (.) bleu (b) de coordonnées (1,-3)
```

Pour $n \in \mathbb{N}^*$, et en vous aidant des commandes précédentes, écrire un programme Python qui trace les points dont les abscisses sont les valeurs de $u_0 \in \llbracket 1, n \rrbracket$ et les ordonnées le temps de vol de la suite de Syracuse de premier terme u_0 .