

Boucles While

Boucle While

On a vu que l'on pouvait utiliser une boucle **for** pour exécuter une série d'instructions un certain nombre de fois. Or, il arrive que l'on ne sache pas à l'avance combien de fois on a besoin d'exécuter cette série d'instructions ! On souhaitera parfois continuer la boucle "tant que ..." ou alors continuer la boucle "jusqu'à ce que ...".

On utilise alors une boucle **while** :

Boucle While

```
while condition :  
    instructions
```

(Tant que la condition **condition** est vraie, on exécute **instructions**)

Remarque 1

Tant que la condition est remplie, on continue de passer dans la boucle. Il est donc nécessaire que les instructions viennent "modifier" cette condition à chaque passage pour qu'elle finisse par être fausse, sans quoi on se retrouve dans une boucle infinie !

Si jamais cela arrive, il faudra interrompre l'exécution du programme (cherchez dans l'onglet "Exécution" ou bien clique droit dans la console...)

Exercice 1

Comprendre le programme suivant et prévoir le résultat qu'il affiche :

```
x=48 ; n=0  
while x > 5 :  
    x=x/2  
    n+=1 # rappel : équivalent à n=n+1  
print('x=',x,'et n=',n)
```

Avant la boucle : $x = \dots$ et $n = \dots$ A-t-on $x > 5$?

Après 1 passage : $x = \dots$ et $n = \dots$ A-t-on $x > 5$?

Après 2 passages : $x = \dots$ et $n = \dots$ A-t-on $x > 5$?

Après 3 passages : $x = \dots$ et $n = \dots$ A-t-on $x > 5$?

Après 4 passages : $x = \dots$ et $n = \dots$ A-t-on $x > 5$?

Texte affiché au final : $x = \dots$ et $n = \dots$

Un exercice très classique consiste à créer un programme qui détermine le premier indice $n \in \mathbb{N}$ tel qu'une certaine propriété $\mathcal{P}(n)$ est réalisée.

```
n = 0 # on initialise avec la première valeur de n  
  
while (not P(n)) : # tant que la condition voulue n'est pas satisfaite...  
    n = n + 1 # ...on augmente la valeur de n  
  
print(n) # quand on sort de la boucle, on affiche ou on renvoie n
```

Exercice 2

On définit : $\forall n \in \mathbb{N}$, $u_n = \frac{2n^2 + 5n + 1}{n^3 + 1}$. On a donc $\lim_{n \rightarrow +\infty} u_n = \dots$

Proposer un programme qui affiche le premier entier $n \in \mathbb{N}$ tel que $u_n \leq \frac{1}{10}$.

Exercice 3

On pose, pour tout $n \in \mathbb{N}^*$, $S_n = \sum_{k=1}^n \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n}$.

On admet dans cet exercice que $\lim_{n \rightarrow +\infty} S_n = +\infty$. Ainsi, par définition de la limite :

Quel que soit $A > 0$, il existe un rang $N_A \in \mathbb{N}^*$ tel que $\forall n \geq N_A$, $S_n > A$.

Compléter le script suivant pour que l'appel de **indice(A)** renvoie le premier indice $n \in \mathbb{N}^*$ tel que $S_n > A$.

```
def indice(A) :  
  
    n = ... ; s = ...  
  
    while ..... :  
  
        n = .....  
        s = .....  
  
    return(.....)
```

- Premier indice tel que $S_n > 0$: • Premier indice tel que $S_n > 3$:
- Premier indice tel que $S_n > 5$: • Premier indice tel que $S_n > 10$:

On peut également adapter cette méthode pour des suites récurrentes.

Exercice 4

Soit $(u_n)_{n \in \mathbb{N}}$ la suite définie par : $u_0 = -1$ et $\forall n \in \mathbb{N}, u_{n+1} = 2\sqrt{u_n + 3}$.

1. On admet que $(u_n)_{n \in \mathbb{N}}$ est croissante et majorée par 6 (récurrence facile).

On note $\ell \in \mathbb{R}$ sa limite. Déterminer (rapidement) la valeur de ℓ .

2. Par définition de la limite, on sait que quel que soit $\varepsilon > 0$,

Il existe $N_\varepsilon \in \mathbb{N}$ tel que $\forall n \geq N_\varepsilon, |u_n - 6| < \varepsilon$.

Compléter la fonction `indice` qui prend en entrée un réel $\varepsilon > 0$ et renvoie le premier rang $n \in \mathbb{N}$ pour lequel $|u_n - 6| < \varepsilon$.

```
def indice(eps) :
    n = ... ; u = ...
    while ..... :
        n = .....
        u = .....
    return(.....)
```

Pour $\varepsilon = 10^{-1}$: on obtient le rang $n = \dots$

Pour $\varepsilon = 10^{-3}$: on obtient le rang $n = \dots$

Pour $\varepsilon = 10^{-10}$: on obtient le rang $n = \dots$

Bonus : le jeu du "plus ou moins"

Exercice 5

On donne les commandes suivantes :

- Après avoir effectué l'importation `import numpy.random as rd`, l'instruction `a = rd.randint(1,101)` stocke dans `a` un entier aléatoire entre 1 et 100.
- L'instruction `b = int(input("mon texte : "))` affiche à l'écran le texte "mon texte :" , attend que l'utilisateur tape une valeur entière dans la console, puis stocke cette valeur dans la variable `b`.

A l'aide de ces commandes, créer un programme Python qui génère un entier aléatoire que l'utilisateur essaiera de deviner. Précisément :

- Le programme génère une valeur aléatoire entre 1 et 100, inconnue de l'utilisateur.
- Le programme demande à l'utilisateur de taper un nombre entre 1 et 100.
- Si l'utilisateur propose une valeur trop élevée, le programme affiche le texte "C'est moins!". Sinon, le programme affiche le texte "C'est plus!".
- Le programme continue d'interroger ainsi l'utilisateur jusqu'à ce qu'il ait deviné la bonne valeur. Il affiche alors un message de victoire !