TP ALGO 5 : LA BOUCLE NON BORNEE WHILE

Pour écrire certains programmes, il est parfois nécessaire de répéter une ou plusieurs instructions un nombre inconnu de fois. On utilise une boucle non bornée qui est parcourue jusqu'à ce qu'une certaine condition ne soit plus vérifiée. Tant que cette condition est vérifiée, la boucle continue.

On parle de « boucle while » ou « boucle non bornée ».

La syntaxe Python pour définir une boucle non bornée est la suivante :

```
while condition:
    instructions
```

Tant que la condition est vraie, on répète les instructions du bloc indenté.

Exemples

- *Que réalise le script ci-contre ? On pourra utiliser un tableau indiquant la valeur de a à chaque passage dans la boucle.*
  Modifier le programme pour qu'il n'affiche que la dernière valeur de a.

```
a = 2
while a <= 50:
    a = a * 3
    print(a)
```

- *Que réalise la fonction définie ci-contre ?*
  On pourra utiliser un tableau et le cas $n=20$ par exemple.

```
def seuil(n):
    S = 3
    while S <= n:
        S = 2*S
    return S
```

Exercices d'application

Exercice 1 : Décrire, comme dans l'exemple 1 ci-dessus, ce qui se passe lors de l'exécution « à la main » du script ci-contre.

```
a = 1
while a < 11:
    a = a + 4
    print(a)
```

Exercice 2 : Avant exécution du script ci-contre, n a pour valeur 0 et P a pour valeur 1.

```
while P < 100:
    n = n + 1
    P = P * 5
```

1. On exécute le script à la main. Compléter le tableau suivant :

n	0	1		
P	1			
P<100	True			

2. On dit que la variable n est un compteur. Pourquoi ?

Exercice 3 : Ecrire une fonction premier(n) qui renvoie True si l'entier n est premier, et False sinon.

Exercice 4 : Dans une culture bactérienne constituée de N bactéries, on suppose qu'à chaque seconde, 2 % des bactéries meurent.

1. Ecrire une fonction $bacteries(N,n)$ qui renvoie le nombre de bactéries encore vivantes après n secondes, n entier naturel.

2. Une culture contient 1 million de bactéries. Combien de bactéries sont encore vivantes après 10 secondes ? Arrondir le résultat à l'unité.

3. En écrivant une fonction appelée $moitié(N)$, déterminer le nombre de secondes qu'il faut au minimum pour que le nombre de bactéries soit strictement inférieur à la moitié de la population initiale. Ce temps dépend-il du nombre de bactéries au départ ?

Exercice 5 : on souhaite écrire un programme dessinant une spirale ayant l'allure ci-contre. Pour être facilement dessinée et néanmoins harmonieuse, cette spirale est constituée de demi-cercles dont les dimensions augmentent régulièrement : chaque demi-cercle a une épaisseur supérieure de 1 à celle du demi-cercle précédent, et un diamètre qui est le carré de cette valeur.

On a donc un programme de construction de ce type :

```
from turtle import *

width(2)
circle(4, 180)
width(3)
circle(9, 180)
width(4)
circle(16, 180)
```

Ecrire un programme (le plus simple possible), demandant à l'utilisateur le nombre de tours de spirale souhaité, et traçant cette spirale.

Ecrire un programme (le plus simple possible), demandant à l'utilisateur la taille maximale de spirale souhaitée, et traçant cette spirale.

Exercice 6 : La conjecture de Syracuse est non encore démontrée à ce jour. Elle s'énonce comme suit : « Soit n un entier naturel. Si n est pair, on le divise par 2, sinon on le multiplie par 3 et on ajoute 1. On recommence ce processus avec le résultat obtenu. La conjecture de Syracuse affirme que l'on obtient toujours 1 au bout d'un nombre fini d'itérations.

1. Ecrire une fonction $Syracuse(n)$ qui répond à la conjecture et la tester pour plusieurs valeurs de n.

2. Modifier cette fonction afin de compter le nombre d'itérations nécessaires jusqu'à l'obtention de 1. Ce nombre d'itérations s'appelle le temps de vol de n.

3. Soit N un entier donné. Ecrire une fonction $maximum(N)$ qui renvoie l'entier n entre 1 et N pour lequel le temps de vol est le plus grand, et le temps de vol correspondant.