Programmation en Python : Boucles
Seconde, 2019-2020

1. La boucle `While`

1.1. Syntaxe

Une boucle non bornée (ou conditionnelle) est la répétition d'une séquence d'instructions, soumise à une condition. Tant que cette condition est vérifiée, la séquence est répétée.
En Python, pour exécuter une ou des instruction(s) **tant qu'une condition est vraie**, on utilise une boucle `While`.

Synthaxe en Python
`While` condition : instruction(s)

Exemple 1.

```
1 entier=0
2 while entier<6:
3     print(entier)
4     entier=entier+1
```

Remarque 1. On déclare la variable `entier` et on l'initialise avec la valeur 0 avant la boucle.

Remarque 2. L'indentation, c'est-à-dire le décalage du texte vers la droite, indique quelles sont les instructions qui sont dans cette boucle.

☞ Exécuter le code ci-dessus. Quels sont les nombres affichés dans la console ? 0 1 2 3 4 5.

☞ Modifier le code pour qu'il affiche tous les entiers de 4 à 12.

```
1 entier=4
2 while entier<13:
3     print(entier)
4     entier=entier+1
```

1.2. Activité 1

On dispose d'un dé parfaitement équilibré. Un jeu consiste à lancer le dé jusqu'à obtenir un 6. Le nombre de lancers qu'il a fallu faire pour obtenir un 6 rapporte la quantité équivalente en bonbons. On souhaite programmer une simulation de ce jeu.
On a écrit le script incomplet ci-dessous qui simule le jeu.
La fonction `randint(a,b)` renvoie un nombre entier aléatoire entre a et b. Elle doit être importée depuis la bibliothèque `random`.

```
1 from random import randint
2 def jeu():
3     lancers=1
4     de=randint(1,6)
5     while de<6:
6         de=randint(1,6)
7         lancers=lancers+1
8     return lancers
```

① Compléter ce programme pour que le jeu fonctionne.

② Recopier ce script dans l'éditeur du logiciel EduPython et le tester plusieurs fois.

③ Quand la boucle s'arrêtera-t-elle ?
la boucle s'arrête lorsque la variable `de` vaut 6.

④ Expliquer le rôle de l'instruction de la ligne 7.
l'instruction de la ligne 7 permet de compter le nombre de fois où l'on rentre dans la boucle `while`, c'est-à-dire le nombre de lancers que l'on va effectuer avant d'obtenir un 6.

2. La boucle For

2.1. Syntaxe

Une boucle bornée permet la répétition d'une séquence d'instructions un nombre fini de fois.
En Python, pour répéter n fois une ou des instruction(s), on utilise une boucle `for`. La variable indiquée prend alors automatiquement toutes les valeurs entières de 0 à n - 1.

Synthaxe en Python
`for` variable `in range(n)` : instruction(s)

Exemple 2.

```
for entier in range(10):
    print(entier)
```

Remarque 3. La variable `entier` est déclarée directement dans la boucle `for` et initialisée avec la valeur 0.

☞ Exécuter le code ci-dessus. Quels sont les nombres affichés dans la console ? 0 1 2 3 4 5 6 7 8 9

☞ Modifier le code pour qu'il affiche tous les entiers de 5 à 20.

```
for entier in range(16):
    print(entier+5)
```

2.2. Activité 2

☞ Compléter le code ci-dessous pour qu'elle renvoie la somme de tous les entiers de 0 à un nombre N donné.

```
def somme(N):
    resultat=0
    for entier in range(N):
        resultat=resultat + entier
    return resultat
```

☞ Combien vaut la somme de tous les entiers de 0 à 1 000 ? 500500

La syntaxe indiquée ci-dessous permet de faire prendre à la variable indiquée toutes les valeurs entières de m à n - 1.

Synthaxe en Python
`for` variable `in range(m,n)` : instruction(s)

☞ Modifier la fonction somme écrite précédemment afin qu'elle renvoie la somme de tous les entiers compris entre deux valeurs données.

```
def somme(M,N):
    resultat=0
    for entier in range(M,N):
        resultat=resultat + entier
    return resultat
```

☞ Combien vaut la somme de tous les entiers de 50 à 100 ? 3825

3. Exercices d'applications

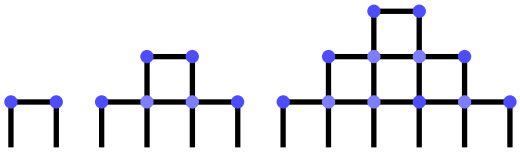
Exercice 1. La croissance du nénuphar

Un nénuphar recouvre 1m^2 d'un étang de 30 m^2. Sachant qu'il double de surface tous les jours, dans combien de jours aura-t-il recouvert tout l'étang ?
Écrire un programme afin de répondre à ce problème.

```
jour=1
surface=1
while surface<=30:
    surface=surface*2
    jour=jour+1
print(jour)
```

EXERCICE 2. Pyramide de baguettes en bois

On souhaite construire des pyramides avec des baguettes en bois de la façon ci-contre :



① On a programmé en langage python la fonction ci-dessous.

```
def pyramide(n):
    b=3
    S=0
    for i in range(n):
        S=S+b
        b=b+4
    return S
```

i	///	0	1	2	3
S	0	3	10	21	36
b	3	7	11	15	19

(a) Compléter le tableau ci-dessus dans le cas où on exécute l'instruction `pyramide(4)`.

(b) Que représentent les différentes valeurs prises par `b` ? `b` est le nombre d'allumettes pour faire un étage supplémentaire.

(c) A quoi correspond le nombre renvoyé par `pyramide(4)` ? le nombre renvoyé par `pyramide(4)` correspond au nombre d'allumettes nécessaires à la construction d'une pyramide de 4 étages .

② On souhaite connaître le nombre maximal d'étages que l'on peut construire avec 500 baguettes en bois.

(a) Compléter le programme ci-contre pour qu'il renvoie le nombre d'étages que l'ont peut construire avec N baguettes en bois.

(b) Modifier le programme pour celui-ci renvoie également le nombre de baguettes en bois restantes.

```
def pyramide(n):
    b=3
    S=0
    for i in range(n):
        S=S+b
        b=b+4
    return S


def nb_etages(N):
    n=0
    while pyramide(n)<=N:
        n=n+1
    return (n-1,N-pyramide(n-1))

print(nb_etages(1000))   # ou bien on peut appeler la
    fonction dans la console
```