Travaux Pratiques 3 : boucle for

En algorithmique, une structure itérative, appelée aussi boucle, sert à répéter une suite d'instructions un certain nombre de fois, de manière automatique. On distingue deux types de boucles :
— lorsque le nombre de répétitions est connu à l'avance, c'est la boucle `for`
— lorsque le nombre de répétitions est conditionné par un objectif à atteindre, c'est la boucle `while`. (cf TP 4).

La syntaxe de la structure `for` est la suivante :
```
for k in range(debut, fin+1, pas):
    actions (indentées)
```

Exercice 1: *introduction à la boucle for*

1. Deviner l'affichage, et l'écrire sur votre feuille avant d'exécuter le programme suivant :
```
for i in range(1,11):
    print(i)
```

2. Même question pour les trois programmes suivants :
```
for i in range(1,21,2):
    print(i)
```
```
for i in range(6):
    print(i**2)
```
```
for i in range(6):
    print(3)
```

Exercice 2:

Deviner ce que renvoie la fonction suivante :
```
def f(n):
    s=0
    for i in range(1,n+1)
        s=s+i
    return s
```
Quelle somme est cachée dans la variable `s` ?

Exercice 3:

Ecrire une fonction qui demande un entier n à l'utilisateur et qui renvoie la valeur $n!$ à l'aide d'une boucle `for`.

Exercice 4: *pour s'entraîner*

Ecrire un programme qui demande un entier n à l'utilisateur et qui affiche les n premiers entiers impairs.
(si l'utilisateur rentre 3, le programme affichera les entiers 1,3 et 5).

Exercice 5: *Suites et boucle for*

1. Deviner l'affichage du programme suivant avant de l'exécuter :
```
u=0
for i in range(1,7):
    u=2*u+3
    print(u)
```

2. Quelle est la suite mathématiques en jeu ?
3. Modifier le programme précédent afin qu'il affiche uniquement u_{100} (mais sans en faire une fonction).
4. Trouver alors la forme explicite de la suite. En déduire un programme qui affiche u_{100} sans utiliser la boucle `for`.

Pour pouvoir étudier davantage de suites, il faut avoir accès aux fonctions usuelles comme la fonction exponentielle, ln, cosinus etc. Toutes ces fonctions usuelles ont été programmées dans python, et ont été regroupées dans une même bibliothèque appelée ***numpy****. Pour pouvoir les utiliser, il faut commencer par importer cette bibliothèque. Plusieurs syntaxes sont possibles (cf cours), mais celle que l'on utilisera pour l'instant est la suivante :*
import numpy as np *(à mettre en première ligne de votre script, ou dans la console si vous n'utilisez pas l'éditeur).*

Exercice 6:

1. Soit la suite u définie par : $u_0 = 0.1$ et $\forall n \in \mathbb{N}$, $u_{n+1} = 1 - e^{-2*u_n}$.
Ecrire une fonction qui au paramètre d'entrée n renvoie la valeur de u_n (pour vérifier : $u_5 = 0.6976037$).
La commande pour la fonction exponentielle est : `np.exp()`
Le préfixe `np` indique à python qu'il doit la chercher dans la bibliothèque `numpy`)

Mêmes questions pour les deux suites suivantes :
2. $u_1 = 0$ et $\forall n \in \mathbb{N}^*$, $u_{n+1} = \frac{1}{2}u_n^2 + 1$ (pour vérifier : $u_8 = 219.13444$)
3. $u_1 = 1$ et $\forall n \in \mathbb{N}^*$, $u_{n+1} = 2nu_n + 3$ (pour vérifier : $u_7 = 135759$).

Exercice 7:

Soit le script suivant (ne pas le taper, ni l'exécuter !) :

```
n=int(input('n?'))
u=1 # u=u_0
v=0 # v=u_1
for i in range(2,n+1):
    w=2*u-3*v
    u=v
    v=w
```

	w	u	v
avant for		1	0
i=2			
i=			

1. Compléter le tableau ci-dessus en y insérant les valeurs successivement prises par `u,v,w` et `i` lorsque `n=3`.
2. Réaliser que les différentes variables informatiques ne cachent que des termes d'une même suite u. Quelle est la relation de récurrence de la suite u ?
3. Quelle variable faut-il afficher à la fin pour obtenir la valeur de u_n ?

Exercice 8: *Suite de Fibonacci*

Soit u la suite définie par $u_0 = 1$ et $u_1 = 1$ et pour tout $n \in \mathbb{N}$, $u_{n+2} = u_n + u_{n+1}$.
Ecrire une fonction de paramètre d'entrée n qui renvoie la valeur de u_n. (pour vérifier : $u_{10} = 89$)

Exercice 9: *Sommes et boucle for*

1. On pose $S_n = \sum_{k=1}^{n} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + .. + \frac{1}{n}$. Ecrire une fonction qui à un entier n renvoie la valeur de S_n.
   Pour vérifier : avec $n = 6$, $S_6 = 2.45$.
2. Conjecturer à l'aide de python $\lim\limits_{n\to+\infty} \frac{S_n}{n}$ ainsi que $\lim\limits_{n\to+\infty} \frac{S_n}{\ln n}$. En déduire $\lim\limits_{n\to+\infty} S_n$.
3. Ecrire une fonction qui à deux entiers n et p renvoie la valeur de $\sum_{k=1}^{n} k^p$ (pour vérifier : $\sum_{k=1}^{10} k^2 = 385$).

Exercice 10: *On mélange ...*

Soit la suite u définie par $u_0 = 14$ et pour tout $n \in \mathbb{N}$, $u_{n+1} = 6 - \frac{1}{2}u_n$.
On pose pour tout $n \in \mathbb{N}^*$, $S_n = \sum_{k=1}^{n} u_k$ la somme des n premiers termes de la suite.

1. Ecrire une fonction python qui au paramètre d'entrée n renvoie la valeur de u_n. (pour vérifier : $u_9 = 3.98046875$)
2. Compléter la fonction précédente, afin qu'elle calcule et renvoie également la valeur de S_n. (pour vérifier : $S_9 = 32.66015625$)
3. *bonus :* trouver la forme explicite de u_n puis de S_n afin d'écrire une variante de la fonction de la question 2. sans boucle `for`.

Exercice 11: *On complique ...*

Reprendre les 3 questions de l'exercice précédent lorsque la suite u est définie par : $u_1 = 0$, $u_2 = -9$ et la relation de récurrence : $\forall n \in \mathbb{N}^*$, $u_{n+2} = 6u_{n+1} - 9u_n$.

Pour vérifier : $u_6 = -3645$, et $S_6 = -4923$

Exercice 12: *Pour finir*

Ecrire une fonction python qui à un entier n associe la valeur de u_n où la suite u est définie par récurrence par : $u_1 = 0$ et $u_2 = 1$ et pour tout entier $n \in \mathbb{N}^*$, $u_{n+2} = (n+1)u_{n+1} - u_n$ (pour vérifier $u_6 = 85$).