BOUCLES FOR

L'essentiel

- Lorsqu'on connaît à l'avance le nombre de répétitions, on peut utiliser une boucle **for** pour gagner du temps :

Lorsque `pas` > 0,

```
i = debut
while i < fin :
        INSTRUCTIONS
        i = i + pas
SUITE
```

$\Longleftrightarrow$

```
for i in range(debut, fin, pas) :
        INSTRUCTIONS
SUITE
```

- S'il n'est pas précisé, le `debut` vaut 0, et le `pas` vaut 1. Quelques exemples :

◇
```
for i in range(2, 6):
    print(i, end=";")
```
affiche `2;3;4;5;`

◇
```
for i in range(10):
    print(i, end=";")
```
affiche `0;1;2;3;4;5;6;7;8;9;`

◇
```
for i in range(2, 11, 3):
    print(i, end=";")
```
affiche `2;5;8;`

◇
```
for i in range(12, 4):
    print(i, end=";")
```
n'affiche rien (ne rentre pas dans la boucle)

◇
```
for i in range(12, 4, -1):
    print(i, end=";")
```
affiche `12;11;10;9;8;7;6;5;`

- Attention : pensez que la valeur de fin n'est jamais atteinte !

Exercices du jour

Exercice 1: On considère les fonctions suivantes qui reçoivent en paramètre un entier naturel n et renvoie la valeur de u_n où (u_n) est une suite à déterminer.

```
def suite1(n):
    u = 8
    for k in range(n):
        u = u/4+2
    return u

def suite2(n):
    return n/4+3

def suite3(n):
    u = 48
    for k in range(n):
        u = u/4+k
    return u
```

1. Écrire les 3 expressions mathématiques des 3 suites correspondant aux codes des 3 fonctions.
2. Vérifier vos résultats par le calcul des premiers termes de chacune des 3 suites.
3. Les résultats changent -ils si on écrit k in range(1,n+1) au lieu de range(n) dans les cas 1 et 3 ?

Exercice 2: On lance un dé n fois de suite et on veut compter combien de fois on tombe sur 6.

1. Créer une fonction `experience6` qui prend en entrée le nombre n de lancers à effectuer et détermine le nombre de 6 obtenus.
2. Modifier la fonction précédente pour qu'elle renvoie la fréquence d'apparition du 6.

Exercice 3: Soit (u_n) la suite définie par : $\begin{cases} u_0 = \dfrac{1}{2} \\ \forall n \in \mathbb{N}, u_{n+1} = 3u_n + 2 \end{cases}$

1. Rappeler comment on peut réaliser une fonction récursive de paramètre `n` permettant de calculer le `n`-ième terme de la suite (cf Séance 7 sur les fonctions récursives).
2. Créer une fonction de paramètre `n` faisant la même chose en utilisant cette fois-ci une boucle `for`.

Exercices en autonomie

Exercice 4: On désire calculer le $n-$ième terme de la suite définie par récurrence par : $\begin{cases} F_0 = 0, \quad F_1 = 1 \\ \forall n \in \mathbb{N}, F_{n+2} = F_{n+1} + nF_n \end{cases}$

1. Complétez la fonction suivante afin que celle-ci renvoie F_n.

```
def SuiteF(n):
    F0 = ...
    F1 . ...
    for i in range(..., ...):
        temp = .......
        F1 = .......
        F0 = .......
    if n == 0:
        return ....
    else:
        return ....
```

2. Des élèves proposent les fonctions suivantes. Lesquelles sont correctes? Lesquelles ne le sont pas et pourquoi? *(Réfléchissez avec de taper pour vérifier vos réponses.)*

Fonction 1

```
def SuiteF1(n):
    F0 = 0
    F1 = 1
    for k in range(n):
        F1, F0 = F1 + n*F0, F1
    return F0
```

Fonction 2

```
def SuiteF2(n):
    F0 = 0
    F1 = 1
    for k in range(n):
        temp = F1
        F1 = F1 + k*F0
        F0 = temp
    return F0
```

Fonction 3

```
def SuiteF3(n):
    F0 = 0
    F1 = 1
    for k in range(n):
        temp = F1
        F1 = F1+(k-1)*F0
        F0=temp
    return F0
```

Exercice 5: On considère la suite $(u_n)_{n\in\mathbb{N}}$ définie par :

$$\begin{cases} u_0 = 1 \\ \forall n \in \mathbb{N},\ u_{n+1} = \dfrac{1}{2}\left(u_n + \dfrac{2}{u_n}\right) \end{cases}$$

Cette suite est la suite de **Héron**, elle constitue une méthode puissante pour obtenir une valeur approchée de $\sqrt{2}$ car elle converge rapidement vers $\sqrt{2}$. C'est d'ailleurs comme ça qu'il y a encore peu de temps les machines calculaient cette valeur.

1. Créer une fonction `Heron` de paramètre `n` permettant de calculer le n-ième terme de la suite de Héron (à l'aide d'une boucle `for`).
2. Créer ensuite une fonction `convergence` prenant en paramètre la précision `eps` et renvoyant le plus petit entier n tel que u_n soit une valeur approchée de $\sqrt{2}$ avec la précision `eps` souhaitée. (On prendra comme valeur de référence de $\sqrt{2}$ la valeur de la bibliothèque `math`.)
3. Déterminer le plus petit terme de la suite de Héron donnant une valeur approchée de $\sqrt{2}$ avec une précision de 10^{-7}. Commenter.

Aide pour les exercices

Solutions des exercices

Correction 4 Sur demande, envoyez un mail à `vincent.maille@ac-amiens.fr`.

Correction 5 Sur demande, envoyez un mail à `vincent.maille@ac-amiens.fr`.