INTRODUCTION A LA PROGRAMMATION PYTHON

#4. Boucle

> **Objectifs:**
>
> - Introduire le concept de boucle bornée et non bornée
> - Programmer des boucles bornées et non bornées

On a souvent besoin dans un programme de répéter l'exécution d'instructions, on utilise pour cela des **boucles**.

Si on connait à l'avance le nombre de répétitions, on utilise une ***boucle bornée***, en Python avec l'instruction `for`
Si on ne connait pas à l'avance le nombre de répétitions, on utilise une ***boucle non bornée***, en Python avec l'instruction `while`

Boucle bornée: for

Voici un exemple de boucle bornée:

```
for i in range(3):      # on répète trois fois
    print(i)            # cette instruction
    print("hello")      # et cette instruction
print("on est sorti du for")
```

```
0
hello
1
hello
2
hello
on est sorti du for
```

L'***itérateur*** `i` va prendre à chaque tour de boucle une valeur de l'ensemble `range(3)`
L'ensemble `range(3)` est un ensemble ordonné de 3 entiers, ***commençant par 0*** et avec un pas de 1 soit: (0 ; 1 ; 2)

Donc:

- au 1er tour de boucle: i=0
- au 2ème tout de boucle: i=1
- au 3ème tour de boucle: i=2

Ensuite, comme l'itérateur a parcouru tout l'ensemble, on sort de la boucle et on passe à la suite du programme le cas échéant.

On peut aussi mettre d'autres paramètres au `range` pour modifier la valeur de départ et le pas:

```
range(0,3) # valeur initiale(inclue):0 ; valeur maximale(exclue):3 ; pas:1 ; résultat:(0;1;2)
           # = range(3)

range(1,3) # valeur initiale(inclue):1 ; valeur maximale(exclue):3 ; pas:1 ; résultat:(1;2)

range(1,7,2) # valeur initiale(inclue):1 ; valeur maximale(exclue):7 ; pas:2 ; résultat:(1;3;5)
```

Mission 4.1.

> Ecrire un programme qui demande à l'utilisateur un nombre entier **n** puis affiche **n** fois votre nom (chacun sur une ligne)

Mission 4.2.

> Ecrire un programme qui demande à l'utilisateur un nombre entier et affiche tous ses diviseurs.
> *Rappel: l'opération* `a % b` *donne le reste de la division euclidienne de a par b*

Boucle non bornée: while

Voici un exemple de boucle non bornée:

```
i = 0
while i < 3:            # tant que la condition (i < 3) est réalisée
    print(i)            # on exécute cette instruction
    i = i + 1           # et cette instruction
print("on est sorti du while")
```

```
0
1
2
on est sorti du while
```

Les instructions dans le `while` sont exécutée tant que la condition du `while` est évaluée à `True`.

Donc:

- au 1er tour: i=0 donc (i < 3) est vrai donc on rentre dans la boucle (qui va incrémenter i)
- au 2ème tour: i=1 donc (i < 3) est vrai donc on rentre dans la boucle (qui va incrémenter i)
- au 3ème tour: i=2 donc (i < 3) est vrai donc on rentre dans la boucle (qui va incrémenter i)

Ensuite, i=3 donc (i < 3) n'est plus vérifiée donc on ne rentre pas dans le `while` et on passe à la suite du programme le cas échéant.

Mission 4.3.

> On reprend la mission 4.1 mais cette fois-ci, on utilisera une boucle non bornée.

Mission 4.4.

> Programmez un jeu où l'utilisateur doit trouver un nombre entier défini par votre programme.
>
> Quand l'utilisateur saisie un nombre:
>
> > - si ce n'est pas le bon nombre:
> >
> > 1. le programme indique à l'utilisateur si le bon nombre est plus grand ou plus petit
> > 2. redemande à l'utilisateur de saisir un nombre
> >
> > - si c'est le bon nombre
> >
> > 1. le programme affiche un message de félicitation
> > 2. le programme se termine

Vidéo

>