NSI | 03

LES BOUCLES

[Stéphane BEAUDET – Frédéric PEURIERE]

Connaître la syntaxe d'une boucle for, savoir utiliser la fonction range(), parcourir une chaine de caractère, utiliser des instructions conditionnelles au sein d'une boucle, savoir utiliser les instructions break et continue.

Connaître la syntaxe et la spécificité d'une boucle while.

1. LA BOUCLE BORNEE FOR (POUR):

✓ **ACTIVITE:** Dans la console, observez les résultats obtenus puis complétez le tableau « aide mémoire ».

Itération avec des nombres entiers: i est de type entier (int)
Le deuxième argument de la fonction print: *end=" "* permet d´éviter un retour à la ligne à chaque tour de boucle en séparant les valeurs affichées par un espace.

```
>>> for i in range(4):
      print(i, end=" ")


La variable i prend
successivement les valeurs 0,
1, 2 et 3.
```

```
>>> for i in range(4):
      print("boucle", end=" ")


Le programme écrit 4 fois le mot
boucle.
```

La lettre i est la plus couramment utilisée mais on peut utiliser un autre nom. Ses affectations successives sont assurées par la fonction range().

```
>>> for i in range (5,10):
      print(i, end=" ")

5 6 7 8 9
```

```
>>> for i in range (0,100,10):
      print(i, end=" ")

De 0 à 90 par pas de 10
```

Itération dans une chaine (inutile de réaffecter la variable *chaine* à chaque fois):
i est maintenant de type chaine (str)

```
>>> chaine="Boucle for"
>>> for i in chaine:
      print(i)

i prend successivement la valeur de chaque caractère de la chaine
```

Actions sur le déroulement de la boucle: break et **continue**

```
>>> chaine="Boucle for"
>>> for i in chaine:
      if i=="f":
        break
      print(i,end="")


Lorsque la condition est
réalisée, le programme sort de
la boucle.
```

```
>>> chaine="Boucle for"
>>> for i in chaine:
      if i=="c" or i=="f":
        continue
      print(i,end="")

Lorsque la condition est
réalisée, le programme ignore ce
tour de boucle et passe au
suivant
```

✓ AIDE MEMOIRE, BOUCLE FOR:

Itération avec des nombres: l´instruction **range()**	
Pour :	Je tape dans la console :
pour générer une série de nombre entiers de 0 à 10	>>> range ( )
pour générer une série de nombre entiers de 5 à 10	>>> range (,)
pour générer une série de 0 à 100 par pas de 5	>>> range (, ,)
Itération avec des nombres et des chaines:	
Ecrire sans retour à la ligne les chiffres pairs compris entre 0 et 10 inclus (0 est un nombre pair)	>>> ..... □□ print (i, end=" ")

Ecrire dix fois le mot *boucle* avec retour à la ligne	>>> mot="boucle"
	>>> .....
	□□ print (mot)

Ecrire les lettres du mot *boucle* séparées par un espace	>>> mot="**boucle**"
	>>> .....
	□□ print (mot)

Action sur les boucles:	
Pour :	J´utilise à l´intérieur d´un bloc if l´instruction:
	break
	continue

2. LA BOUCLE NON BORNEE WHILE (TANT QUE):

A la différence de la boucle FOR, avec WHILE le programme n´entre dans la boucle que si la condition d´entrée est vraie sinon elle est ignorée. Il faut toujours prévoir une sortie de la boucle, sinon la boucle devient infinie. Si par mégarde cela vous arrive, pensez à stopper l´exécution du programme avec l´icône 

✓ ACTIVITE:

```
>>> i=1
>>> while i<=30:
      print(i,end=".")
      i+=1

Le programme entre la boucle et l´exécute tant que la condition est
vraie. Lorsqu´elle devient fausse, le programme sort de la boucle.
Sans l´incrémentation de la variable i, la boucle serait infinie.
```

✓ AIDE MEMOIRE, STRUCTURE DES BOUCLES:

Pour:	Je tape en Python:
répéter 9 fois les instructions ce qui s´écrit en pseudo code: pour i allant de 0 à 9 *instructions* fin du pour	..... □□□□ instructions
répéter une instruction tant que la condition est vérifiée, ce qui s´écrit en pseudo code: tant que *condition* *instructions* fin du tant que	..... □□□□ instructions