Python la boucle while

Boucle non bornée, Indentation, itération, Condition

La boucle while en résumé

- Dans un programme, on peut avoir besoin de répéter des actions similaires (calculs, affichages, etc.) un nombre inconnu de fois. On utilise alors une instruction appelée boucle non bornée ou boucle while.

L'instruction while est une instruction composée, c'est-à-dire une instruction dont l'en-tête se termine par deux-points :après la ou les condition à respecter, suivie d'un bloc indenté qui constitue le **corps de la boucle**.

On dit que l'on réalise une **itération** de la boucle à chaque fois que le corps de la boucle est exécuté.

Dans l'en-tête de la boucle, on précise après le mot-clé while et la condition à respecter qui répétera les instructions autant de fois nécessaire tant que la condition est respectée.

La syntaxe pour écrire une boucle while non bornée en langage Python est :

```
while <conditions>:
    <Instructions>
```

Pour déterminer le nombre de fois où l'instruction est réalisée, il suffit de créer une variable *i* initialisée à 0 à laquelle on ajoute 1 à chaque fois que les instructions sont réalisées. Cette variable est appelée compteur. La syntaxe en langage Python est :

```
i=0
while <conditions>:
    i=i+1
    <Instructions>
```

Boucle while :

a. Principe

Dans un programme, on peut avoir besoin de répéter des actions similaires (calculs, affichages, etc.) un nombre inconnu de fois. On utilise alors une instruction appelée **boucle non bornée** ou **boucle while**.

La syntaxe pour écrire une boucle while non bornée est :

Langage naturel	Langage Python
Tant que <conditions> <Instructions> Fin Tant que	while <conditions>: <Instructions>

Remarques

Avec une boucle while, on ne peut pas toujours prévoir le nombre de fois où les instructions seront réalisées pour avoir la réponse attendue, contrairement aux boucles for.

Tant que la condition est vérifiée, l'instruction est exécutée.

b. Exemple

Langage naturel	Langage Python

$a \leftarrow 0$
Tant que $a < 132$
  $a \leftarrow a + 7$
Fin Tant que
	`a=0`
	`while a<132:`
	`  a=a+7`
	`print(a)`

La boucle permet d'ajouter le nombre 7 à la variable *a* tant qu'elle est strictement inférieure à 132. On peut résumer les étapes dans le tableau ci-dessous :

Étapes	1	2	...	18	19
Valeurs prises par *a*	0	7	...	126	133

En fait, ce programme cherche le plus petit multiple de 7 supérieur ou égal à 132. Le résultat affiché est alors 133.

Remarque

Voici les erreurs de saisie les plus classiques :

```
while a<132
    a=a+7
```

Oublier les deux points.

```
while a>=132:
    a=a+7
```

Se tromper sur la condition. Pour avoir le plus petit multiple de 7 **supérieur ou égal à 132**, il est nécessaire de calculer les termes tant qu'ils sont **strictement inférieurs à 132**.

```
while a<132:
a=a+7
```

Oublier ou effacer par mégarde la tabulation (indentation) de la ligne 3.

Utilisation de la boucle while :

a. Création d'un compteur

Il est parfois utile de créer une variable qui détermine le nombre de fois où l'instruction est réalisée.

Pour cela, il suffit de créer une variable *i* initialisée à 0 à laquelle on ajoute 1 à chaque fois que les instructions sont réalisées. Cette variable est appelée **compteur**.

La syntaxe est la suivante :

Langage naturel	Langage Python

$i \leftarrow 0$
Tant que <conditions> faire
  $i \leftarrow i + 1$
  <Instructions>
Fin Tant que
	i=0
	while <conditions>:
	  i=*i*+1
	  <Instructions>

Remarques

Nous pouvons simplifier l'écriture i=i+1 par i +=1

Exemple

Langage naturel	Langage Python

$i \leftarrow 0$
$a \leftarrow 0$
Tant que $a < 16$ faire
  $i \leftarrow i + 1$
  $a \leftarrow a + 3$
Fin Tant que
	`i=0`
	`a=0`
	`while a<16:`
	`    i=i+1`
	`    a=a+3`

On peut résumer les étapes dans le tableau ci-dessous :

Étapes	1	2	3	4	**5**	6
Valeurs prises par i	0	1	2	3	4	**5**	6
Valeurs prises par a	0	3	6	9	12	**15**	18

La boucle permet d'effectuer 6 fois les instructions. La variable i prend la valeur 6.

b. Choix des conditions et affichage des résultats

L'algorithme de l'exemple précédent permet de répondre à deux questions différentes :

- Quel est le plus petit multiple de 3 supérieur ou égal à 16 ?

On remarque que pour trouver ce multiple supérieur ou égal à 16, on calcule tous les multiples de 3 tant qu'ils sont strictement inférieurs à 16.
Pour afficher la réponse, on affiche la valeur prise par la variable a grâce à l'instruction print(a).

```
i=0
a=0
while a<16:
    i=i+1
    a=a+3
print(a)
```

Remarque
Le compteur i n'est pas utile ici, sauf pour la curiosité de savoir en combien de fois la réponse a été obtenue.