

TD 4 : LES BOUCLES BORNÉES

A) Qu'est-ce qu'une boucle bornée ?

Définition : Dans un algorithme, une **boucle** est une suite d'instructions que l'on répète (en boucle) un certain nombre de fois.

On dit que la boucle est **bornée** lorsque l'on sait exactement combien de fois on va répéter la boucle.

Exemple n° 1 : On souhaite obtenir la table de multiplication par 3 à l'aide d'un algorithme. On sait que :

$3 \times 0 = 0$	$3 \times 4 = 12$	$3 \times 8 = 24$
$3 \times 1 = 3$	$3 \times 5 = 15$	$3 \times 9 = 27$
$3 \times 2 = 6$	$3 \times 6 = 18$	$3 \times 10 = 30$
$3 \times 3 = 9$	$3 \times 7 = 21$	$3 \times 11 = 33$

On remarque que l'opération effectuée est toujours la même : $3 \times i$ où i est un entier compris entre 0 et 11. Voici alors l'algorithme et le programme associé :

Algorithme	Python
Pour i allant de 0 à 11 faire : Afficher $3 \times i$ Fin Pour	<pre>For i in range(12): print(3 * i)</pre>

Dans l'algorithme, les instructions qui seront répétées sont celles comprises entre la ligne "Pour" et la ligne "Fin Pour".

Comme pour les instructions conditionnelles, l'indentation est obligatoire avec Python. La commande "range" est expliquée dans la partie B.

i est appelé un compteur. Au début de l'algorithme, la valeur du compteur i est 0 donc on exécute "Afficher 3×0 ". Puis, la valeur de i devient 1 et alors, on exécute "Afficher 3×1 ". Et ainsi de suite jusqu'à ce que i soit égal à 11. Dans ce cas, on exécute "Afficher 3×11 " puis l'algorithme s'arrête.

Info : Les boucles bornées utilisées au lycée peuvent parfois être appelées les "boucles Pour" ou les "boucles For" en lien avec les mots-clés utilisés dans les algorithmes et dans les programmes avec Python.

B) Utilisation de l'instruction "range"

On utilise l'instruction "range" lorsque l'on sait exactement combien d'itérations de la boucle on souhaite effectuer. De plus, le départ de la boucle est toujours 0 dans ce cas. Dans notre exemple, i commence à 0 et va jusqu'à 11 : on utilise donc 12 valeurs de i et c'est pour cela que l'on écrit "range(12)".

Si on veut commencer la table de multiplication à 3×1 et finir à 3×10 , on utilisera alors la syntaxe suivante.

Exemple n°2

Algorithme	Python
Pour i allant de 1 à 10 faire : Afficher $3 \times i$ Fin Pour	<i>For i in range(1, 11): print(3*i)</i>

L'utilisation de "range" est ici très particulière : "range(a, b)" signifie que a sera la première valeur. En revanche, on doit calculer $b - a$ pour connaître le nombre d'itération. Dans notre cas, $11 - 1 = 10$ itérations de la boucle.

Si on ne souhaite afficher que les multiples pairs, on pourra préciser un pas.

Exemple n°3

Algorithme	Python
Pour i allant de 0 à 10 avec un pas de 2 faire : Afficher $3 \times i$ Fin Pour	<i>For i in range(0, 11, 2): print(3*i)</i>

Pour commencer, i sera égal à 0, puis à 2, puis 4, et ainsi de suite jusqu'à 10.

On peut également utiliser une boucle bornée lorsque l'on connaît exactement les valeurs du compteur. On utilise pour cela la notation "[x_1, x_2, ..., x_n]" où $x_1, x_2, \dots, x_n$ sont les valeurs que va prendre le compteur.

Avec cette notation, les boucles bornées peuvent être utiles pour réaliser un tableau de valeurs d'une fonction par exemple.

Exemple n° 4

Algorithme	Python <i>l'int 2</i>
Fonction f(x): Retourner Pour i dans {-2 ; -0,5 ; 0 ; 0,25 ; 1} faire: Afficher f(i) Fin Pour	<pre> <i>l'int 2</i> for i in range (-2, 0.5, 0.25): print(f(i)) </pre>

À retenir :

range(n) : effectue une boucle avec n itérations en partant de 0 jusqu'à n - 1.

range(a, b) : effectue une boucle avec b - a itérations en partant de a.

range(a, n, p) : effectue une boucle en partant de a jusqu'à n - 1. La valeur du compteur est augmentée de p à chaque itération.

[x_1, x_2, ..., x_n] : effectue une boucle avec n itérations en prenant successivement les valeurs x_1, x_2, ..., x_n.

C) Exercices

Exercice 1 :

- 1) Écrire un programme qui affiche la table de multiplication par 4 depuis 4×0 jusqu'à 4×10 .
- 2) Modifier ce programme pour qu'il affiche la table de multiplication par 4 depuis 4×3 jusqu'à 4×8 .
- 3) Écrire une fonction `TableMultiplication(n)` qui affiche la table de multiplication par n depuis $n \times 0$ jusqu'à $n \times 10$. Quel est l'intérêt d'écrire une telle fonction ?

Exercice 2 :

permet d'afficher n importe quelle table de mult..

On considère la fonction f définie sur $\mathbb{R}$ par $f(x) = x^2$.

- 1) Définir cette fonction avec Python.
- 2) Écrire un programme qui affiche les images des entiers entre -3 et 3 par la fonction f .
- 3) Modifier ce programme pour n'afficher que les images des entiers pairs entre -3 et 3.

Exercice 3 :

Les boucles permettent à un compteur de parcourir une chaîne de caractères. On considère l'exemple suivant.

```
1 mot = "MATHEMATIQUES"
2 n = 0
3 for i in mot:
4     if i == "A":
5         n = n + 1
6 print(n)
```

- 1) Quel est le type de la variable mot ? *string*
- 2) Quelles vont être les valeurs parcourues par i à chaque passage de la boucle "for" ?
- 3) Ecris un programme qui affiche le nombre de lettres A puis de lettres N contenu dans le mot français le plus long :

INTERGOUVERNEMENTALISATIONS

Exercice 4 :

On considère le programme suivant où deux boucles "for" sont emboîtées :

```
1 for i in range(5):
2     for j in range(5):
3         print(i + j)
```

- 1) Quelle est la première valeur des compteurs i et j au début du programme ? *0 et 0*
- 2) Quelle est alors la première valeur affichée par le programme ? *la somme de i + j pour chaque paire de valeurs*
- 3) Après l'affichage de la première valeur, quel compteur va changer en premier : i ou j ? Quelle est alors la deuxième valeur affichée ? *i 1*
- 4) Exécuter ce programme pour visualiser les résultats.

Exercice 5 :

On considère la chaîne de caractères "AEIOUY" représentant les voyelles de l'alphabet français.

En s'aidant des exercices 3 et 4, écrire un programme qui compte le nombre total de voyelles dans un mot donné (on pourra utiliser les mots "MATHEMATIQUES" et "INTERGOUVERNEMENTALISATIONS" en exemples).