

CHAPITRE III : Les boucles

BOUCLE « TANT QUE »

La résolution de certains problèmes conduit parfois à répéter un bloc d'instructions tant qu'une condition est vérifiée (termes d'une suite par exemple). C'est une **itération conditionnelle**.

Algorithme (langage usuel)	Programme (langage Python)
Tant que <i>condition vérifiée</i> faire <i>bloc d'instruction1</i> Fin tant que	while <i>condition vérifiée</i> : <i>bloc d'instruction1</i>

- Les « : » associés sont **obligatoires** et correspondent au mot « faire »
- Le **décalage (indentation)** du bloc d'instruction est **obligatoire** :

Comme pour les tests

Rq :Attention, while peut boucler à l'infinie si la condition est toujours vrai. Attention à son utilisation !!
En cas de plantage de Python à cause d'une boucle infinie, stopper le programme avec **Ctrl F2** .

Ex1 : Testons.

- Deviner ce que fait le petit programme suivant.
.....
- Vérifier
- Le modifier pour afficher les multiples de 11 inférieurs à 200.
- Deviner l'effet de la modification ci-contre.
.....
- Vérifier

```
table=0
while table < 50:
    table=table+5
    print (table)
```

```
table=0
while table < 50:
    table=table+5
    print (table)
```

Ex2 : Tables de multiplications.

Faire un programme qui demande à l'utilisateur un nombre entier et qui affiche sa table de multiplication sous la forme : « 0 x 7 = 0 », puis « 1 x 7 = 7 » etc jusqu'à « 10 x 7 = 70 ».
Modifier votre programme pour que l'utilisateur choisisse jusqu'à combien...

Ex3 : Diviseurs.

Faire un programme qui demande à l'utilisateur un nombre entier et qui affiche tous ses diviseurs.

Ex4 : Nombres premiers.

Faire un programme qui demande à l'utilisateur un nombre entier et qui détermine si il est premier.
(*rappel : un nombre premier est un entier naturel qui admet exactement deux diviseurs distincts entiers (qui sont alors 1 et lui-même). Cette définition exclut 1, qui n'a qu'un seul diviseur entier positif.*)

Ex5 : Nombres premiers (bis)

En réutilisant le programme précédent, afficher les 100 premiers nombres premiers.

Ex6 : Premier jeu (bis)

Reprendre l'Ex8 du chapitre II en proposant de rejouer, ajouter un compteur et afficher « coulé en ... coups ».

LA BOUCLE «POUR»

La fonction **range()** de Python produit une liste d'entiers. On peut préciser un, deux ou trois paramètres : **range(fin)**, **range (début,fin)** et **range (début,fin,pas)**. Tester les instructions suivantes.

```
>>>list( range(10))
.....
# Par défaut range(n) va de 0 à n-1 avec un pas de 1

>>> list(range(270, 0, -30) )
.....
# On va de 270 à 0 exclu. Le pas est ici -30.
```

Dans un problème, on peut connaître à l'avance le nombre d'itérations (tableau de valeurs d'une fonction, ...). Cette une **itération bornée**.

Algorithme (langage usuel)	Programme (langage Python)
Pour i variant de 1 à n faire <i>bloc d'instruction</i> Fin Pour	for i in range (1,$n+1$) : <i>bloc d'instruction</i>

NB : Comme la borne supérieure est exclue, **compter de 1 à n** se fait avec **range(1, $n+1$)**

Rq : La boucle « for » est un cas particulier de boucle « while $i < N$ » avec $i = i+1$ dans le bloc d'instruction.

Ex 7 : Que de nombres !!!

Refaire les exercices 2 à 5 en utilisant la boucle «POUR». Il faudra parfois imbriquer deux boucles.

Ex 8 : Factorielle

Faire un programme qui demande à l'utilisateur un nombre entier et qui détermine sa factorielle. La factorielle d'un entier naturel n est le produit des nombres entiers strictements positifs inférieurs ou égaux à n . Cette opération est notée avec un point d'exclamation, $n!$, et se lit « factorielle n ». Afficher sous la forme (par exemple pour $n = 5$) : « 1 x 2 x 3 x 4 x 5 = 120 »

Ex 9 : Somme de carrés

Faire un programme qui demande à l'utilisateur un nombre entier et qui détermine la somme des carrés des entiers jusqu'à ce nombre.

Afficher sous la forme (par exemple pour $n = 5$) : « $1^2 + 2^2 + 3^2 + 4^2 + 5^2 = 55$ »

Ex 10 : Juste prix (mini projet)

Écrire un programme qui demande de trouver un prix (entier fixé au départ) compris entre 1 et 100 €, affiche « C'est plus » ou « C'est moins » jusqu'à ce que vous ayez « Gagné ! »

- Vérifier que le prix est bien entre 1 et 100, redemander sinon.
- Ajouter : "Gagné en ... coups"
- Limiter à 10 coups, puis donner la réponse et préciser « Perdu ! » si en 10 essais on n'a pas trouvé.
- Proposer de rejouer en fin de partie
- Comptabiliser les parties gagnées, et afficher le nombre de parties gagnées à la fin du jeu.