



## CHAPITRE V : 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 suites par exemple). C'est une **itération conditionnelle**.

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

Comme pour les tests

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

Rq : while(condition) boucle à l'infinie (1 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 : Révisions.

- 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 <=90:
    print (table)
    table=table+9
```

```
table=0
while table <=90:
    print (table)
    table=table+9
```

- Faire un programme qui affiche « 0 fois 11 vaut 0 », puis « 1 fois 11 vaut 11 » etc jusqu'à 198.

#### Ex2 : Puniton.

Faire un programme écrivant 100 fois « Je ne copie pas sur mon voisin », en numérotant les lignes.

#### Ex3 : Rebonds

Une balle tombe d'une hauteur de 2 m et rebondi de 90 % de sa hauteur de chute.

Elle ne rebondit plus si elle chute de 2 cm soit 0,02 m ou moins.

Faire un programme affichant le nombre de rebonds de la balle et la hauteur des rebonds successifs.

 Appeler le professeur pour validation

#### Ex4 : Fibonacci (pour les rapides)

La suite de Fibonacci est définie par  $u_0=1$   $u_1=1$  puis  $u_n=u_{n-1}+u_{n-2}$  pour tout entier  $n \geq 2$ .

Faire afficher la listes des termes de cette suite, inférieurs à 50



## 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
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

>>> list(range(270, 0, -30) )      # On va de 270 à 0 exclu. Le pas est ici -30.
[270, 240, 210, 180, 150, 120, 90, 60, 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)
<b>Pour <math>i</math> variant de 1 à <math>n</math> faire</b> <i>bloc d'instruction</i> <b>Fin Pour</b>	<b>for <math>i</math> in range (1,<math>n+1</math>) :</b> <i>bloc d'instruction</i>

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

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

### Ex 5 : Calendrier

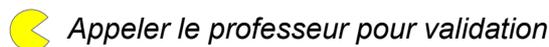
Écrire un programme qui affiche « 1 janvier », jusqu'à « 31 janvier »

### Ex 6 : Tables de multiplications

Écrire un programme qui affiche les tables de multiplications jusqu'à 10 sous la forme « 1 x 1 = 1 » jusqu'à « 10 x 10 = 100 » . Attention il faudra imbriquer deux boucles.

### Ex 7 : Somme de carrés

Faire afficher la liste et la somme des carrés des entiers jusqu'à 100 sous la forme «  $0^2+1^2+2^2+.....+100^2 = .....$  »



### Ex 8 : Premiers (pour les rapides)

Écrire un programme qui affiche les 100 premiers nombres premiers

Indication :  $n\%k$  donne le reste de la division de  $n$  par  $k$ .

Ainsi si  $n\%2$  vaut 0, c'est que  $n$  est pair, et si  $n\%k$  vaut 0 alors  $k$  divise  $n$ .

Article sur les nombres premiers (wikipédia).

« Un nombre premier est un entier naturel qui admet exactement deux diviseurs distincts entiers et positifs (qui sont alors 1 et lui-même). Cette définition exclut 1, qui n'a qu'un seul diviseur entier positif.

Par opposition, un nombre non nul produit de deux nombres entiers différents de 1 est dit composé.

Par exemple  $6 = 2 \times 3$  est composé, tout comme  $21 = 3 \times 7$ , mais 11 est premier car 1 et 11 sont les seuls diviseurs de 11. Les nombres 0 et 1 ne sont ni premiers ni composés. »