TP 3 — Les boucles

I. Boucle non bornée *TANT QUE*

La résolution de certains problèmes conduit parfois à répéter un bloc d'instructions tant qu'une condition est vérifiée (exemple : calcul d'images par une fonction jusqu'à une certaine valeur). C'est une **itération conditionnelle**.

Algorithme (langage usuel)	Programme en langage *Python*
Tant que *condition vérifiée* **faire** bloc d'instruction 1 **Fin Tant que**	**while** condition vérifiée **:** bloc d'instruction 1

On remarque :

- L'indentation (ou décalage) qui permet de délimiter le bloc d'instructions à effectuer ;
- Les deux points **:** qui correspondent au mot "*faire*"

Exercice 1 ***Que fait ce programme ?***

1. Que permet de faire le script ci-dessous ?

```
1  table = 0
2  While table <= 90:
3      print(table)
4      table = table + 9
```

2. Écrire un script qui permet d'afficher les multiples de 11 inférieurs à 200.

3. Deviner l'effet de la modification ci-dessous :

```
1  table = 0
2  While table <= 90:
3      print(table)
4  table = table + 9
```

4. Écrire un programme qui affiche « 0 fois 11 vaut 0 », « 1 fois 11 vaut 11 » etc jusqu'à 198.

Exercice 2 Punition !

Écrire un programme qui écrit 100 fois le texte suivant « Je ne copie pas sur mon voisin », en numérotant les lignes.

Exercice 3 Rebonds

Une balle tombe d'une hauteur de 2 m et rebondit de 90% de sa hauteur de chute.
Elle ne rebondit plus si elle chute de 2 cm (soit 0,02 m) ou moins.
Écrire un programme affichant le nombre de rebonds de la balle et la hauteur des rebonds successifs.

Exercice 4 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 $n \geq 2$.
Écrire un programme affichant la liste des termes de ectte suite, inférieurs à 50.

II. Boucle bornée *POUR*

La fonction **`range()`** de Python produit une liste d'entiers. On peut préciser un, deux ou trois paramètres : **`range(fin)`** ou **`range(début,fin)`** ou **`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(10))     # on va de 270 à 0 exclu. Le pas est de -30
[270,240,210,180,150,120,90,60,30]
```

La résolution de certains problèmes conduit parfois à répéter un bloc d'instructions tant qu'une condition est vérifiée (exemple : tableau de valeurs d'une fonction). C'est une **itération bornée**.

Algorithme (langage usuel)	Programme en langage *Python*
Pour *i* variant de 1 à n faire bloc d'instruction 1 **Fin Pour**	**For i in range(1,n+1):** bloc d'instruction 1

On remarque :

- L'indentation (ou décalage) qui permet de délimiter le bloc d'instructions à effectuer ;
- Les deux points **:** qui correspondent au mot "*faire*" ;
- La borne supérieure étant exclue, pour compter de 1 à n on utilise `range(1,n+1)`.

La boucle ***FOR*** est un cas particulier d'une boucle ***WHILE i < n***.

Exercice 5 Calendrier

Écrire un script qui affiche « 1 janvier » jusqu'à « 31 janvier ».

Exercice 6 Tables de multiplication

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

Exercice 7 Somme de carrés

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

Exercice 8 Nombres premiers

Écrire un programme qui affiche les 100 premiers nombres premiers.

Définition : Un nombre premier est un entier naturel qui admet exactement deux diviseurs distincts entiers et positifs. Ces deux diviseurs sont 1 et le nombre considéré. Par exemple, le nombre entier 7 est premier car 1 et 7 sont les seuls diviseurs entiers et positifs de 7.

Indication : n%k renvoie le reste de la division de n par k.