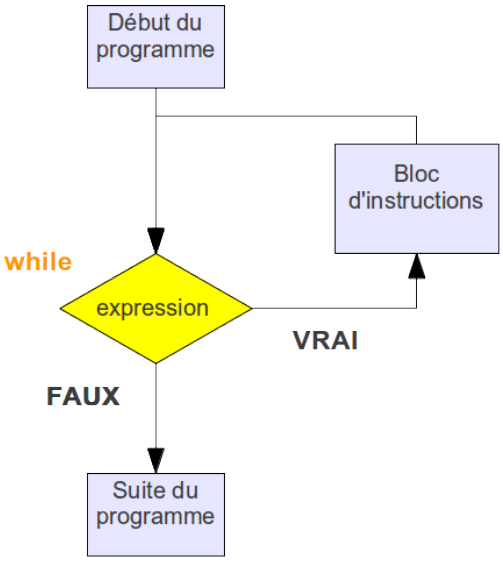
Chapitre 3 - Les boucles

L'instruction `while` *("tant que")*

Une boucle permet d'exécuter une portion de code plusieurs fois de suite.



Syntaxe

```
while expression:
      bloc d'instructions
```

Si l'expression est vraie (`True`) le bloc d'instructions est exécuté, puis l'expression est à nouveau évaluée.
Le cycle continue jusqu'à ce que l'expression soit fausse (`False`) : on passe alors à la suite du programme.

Exemple : un script qui compte de 1 à 4

```
compteur = 1        # initialisation du compteur
while compteur<5:
    print(compteur)
    compteur += 1    # incrémentation du compteur
```

La boucle est répétée *tant que* le compteur est inférieur à 5.

Exemple : Ecrire la table de multiplication par 8 :

```
compteur = 1
while compteur<=10:
    print(compteur, '* 8 =', compteur*8)
    compteur += 1
```

Exemple: Affichage de l'heure courante

```
import time     # importation du module time
quitter = 'n'   # initialisation
while quitter != 'o':
    # ce bloc est exécuté tant que la condition est vraie
    # strftime() est une fonction du module time
    print('Heure courante ', time.strftime('%H:%M:%S'))
    quitter = input("Voulez-vous quitter le programme (o/n) ?
")
print("A bientôt")
>>>
Heure courante  09:48:54
Voulez-vous quitter le programme (o/n) ? n
Heure courante  09:48:58
Voulez-vous quitter le programme (o/n) ? o
A bientôt
```

L'instruction for

Syntaxe

```
for élément in séquence :
    bloc d'instructions
```

Les éléments de la séquence sont issus d'une séquence de nombres, ou chaîne de caractères ou bien d'une liste.

Fonction range()

La fonction `range()` est très utile pour créer des séquences automatiques de nombres entiers : par exemple, `range(1,5)` correspond aux nombres entiers de 1 à 4.

```
print(list(range(1,5)))
for i in range(1,5):
    print(i)
```

La boucle est ici exécutée 4 fois.

Exemple : Ecrire la table de multiplication par 8 est plus facile avec un for plutôt qu'un while :

```
for compteur in range(1,11):
    print(compteur, '* 8 =', compteur*8)
print("Et voilà !")
```

Exemple avec une **séquence de caractères**

```
chaine = 'Bonsoir'
for lettre in chaine:  # lettre est la variable d'itération
    print(lettre)
```

Exemple avec les **éléments d'une liste**

```
maliste = ['Pierre', 67.5, 18]
for element in maliste:
    print(element)
print("Fin de la boucle")
```

Resultat:

```
Pierre
67.5
18
Fin de la boucle
```

Astuce
Si vous connaissez le nombre de boucles à effectuer, ou pour parcourir les éléments d'une chaîne de caractères ou liste, utiliser une boucle `for`.
Autrement, utiliser une boucle `while`, notamment pour faire des boucles sans fin, pour des jeux par exemple.

Des boucles dans les boucles…
Rien n'empêche de placer une boucle à l'intérieur d'une boucle. Par exemple :

```
for compteur1 in range(1,4):
    for compteur2 in range(1,4):
        print(compteur1, compteur2)
```

QCM sur les boucles

Qu'affiche les scripts suivants ?

```
n = 0
while n<15 :
    n = n + 2
print(n)
```

A) 14 B) 15 C) 16 D) 17

```
n = 10
while n>=11 :
    n = n + 2
print(n)
```

A) 10 B) 11 C) 12 D) 13

```
n = 0
for i in range(5) :
    n = n + 1
print(n)
```

A) 4 B) 5 C) 6 D) 7

```
n = 0
for i in range(5) :
    n = n + 1
print(i)
```

A) 4 B) 5 C) 6 D) 7

```
resultat = ""
for c in "Bonsoir" :
    resultat = resultat + c
print(resultat)
```

A) Bonsoir

B) riosnoB

C) BonsoirBonsoirBonsoirBonsoirBonsoirBonsoirBonsoir

Exercices du chapitre 3

Exercice 3.1 ★
Ecrire un script qui demande à l'utilisateur un nombre compris entre 1 et 3 jusqu'à ce que la réponse convienne.

Exercice 3.2 ★
Ecrire un script qui demande un nombre compris entre 10 et 20, jusqu'à ce que la réponse convienne. En cas de réponse supérieure à 20, on fera apparaître un message : « Plus petit ! », et inversement, « Plus grand ! » si le nombre est inférieur à 10.

Exercice 3.3 ★
Ecrire un script qui demande un nombre de départ, et qui ensuite affiche les dix nombres suivants. Par exemple, si l'utilisateur entre le nombre 17, le programme affichera les nombres de 18 à 27.

Exercice 3.4 ★
Ecrire un script qui demande un nombre de départ, et qui calcule la somme des entiers jusqu'à ce nombre. Par exemple, si l'on entre 5, le programme doit calculer :

```
1 + 2 + 3 + 4 + 5 = 15
```

NB : on souhaite afficher uniquement le résultat, pas la décomposition du calcul.

Exercice 3.5 ★★

Ecrire un script qui affiche toutes les tables de multiplication (de 1 à 10).

Exercice 3.6 ★★

Ecrire un script qui calcule la moyenne d'une série de notes.
On pourra utiliser une variable qui contient la somme intermédiaire des notes.

```
>>> Nombre de notes ? 3
-->  15
-->  11.5
-->  14
Moyenne : 13.5
```

Exercice 3.7 ★★
1) Avec une boucle `for`, écrire un script qui compte le nombre de lettres z dans une chaîne de caractères.
Par exemple :

```
>>> Entrer la chaîne : Zinedine Zidane

Résultat : 2
```

2) Faire la même chose, directement avec la méthode `count()` de la classe `str`.
Pour obtenir de l'aide sur cette méthode :

```
>>> help(str.count)
```

Exercice 3.8 ★★ plus grand-plus petit...

1) Ecrire le script du jeu de devinette suivant :

```
>>> Le jeu consiste à deviner un nombre entre 1 et 100 :
--->    50 trop petit !
--->    75 trop petit !
--->    87 trop grand !
--->    81 trop petit !
--->    84 trop petit !
--->    85 Gagné en 6 coups !
```

2) Quelle est la stratégie la plus efficace ?

3) Montrer que l'on peut deviner un nombre en 7 coups maximum.

Bibliographie : La dichotomie

Remarque : pour créer un nombre entier aléatoire entre 1 et 100 :

```
import random

nombre = random.randint(1,100)
```

Exercice 3.9 ★★★ Code de César

En cryptographie, le code de César est une technique de chiffrement élémentaire qui consiste à décaler une lettre de 3 rangs vers la droite :
A → D
B → E
...
Z → C

1) Ecrire le script de ce codage.
Par exemple :

```
>>> Message à coder ? abcdefghijklmnopqrstuvwxyz

defghijklmnopqrstuvwxyzabc

>>> Message à coder ? Monty Python's Flying Circus

prqwb sbwkrq'v ioblqj flufxv
```

On pourra utiliser la chaîne `'abcdefghijklmnopqrstuvwxyz'` et la méthode `find()` de la classe`str`.
Pour obtenir de l'aide sur cette méthode :

```
>>> help(str.find)
```

2) Ecrire le script du décodage.
Par exemple :

```
>>> Message à décoder ?
prqwb sbwkrq'v ioblqj flufxv
monty python's flying circus
```

Exercice 3.10 ★★★

1) Ecrire un script qui détermine si un nombre entier est premier ou pas.
Par exemple :

```
>>> Nombre ? 17
17 est un nombre premier
```

2) Ecrire un script qui décompose un nombre entier en un produit de facteurs premiers.

```
>>> Nombre à décomposer ? 2142
2142 = 2 * 3 * 3 * 7 * 17
```