Fiche 3 : boucle for

Grâce aux structures conditionnelles et à la boucle `while` il est possible de réaliser la plupart des programmes informatiques demandés. Mais il arrive que la syntaxe puisse être lourde. Un cas particulier est donné par un programme du type :

```
n=int(input())  """entier donné par un utilisateur """
compteur=0
while (compteur<n) :
    appliquer l'instruction I
    compteur=compteur+1
```

Dans ce contexte, on constate que l'on effectue n fois l'instruction I et que l'entier `compteur` est utilisé pour vérifier le nombre de fois que l'instruction a été réalisée.

Lorsque l'on connaît par avance le nombre de fois qu'une instruction doit être effectuer, il est plus commode d'utiliser la boucle `for`. Si on cherche à répéter n fois une instruction I, la syntaxe est la suivante :

répéter n fois :	`for _ in range(0,n) :`
Instructions I	Instructions I

Exemple : afficher n fois bonjour

```
for _ in range(0,n) :
    print("bonjour")
```

Exemple : afficher les nombres pairs compris entre 1 et n

```
for i in range(1,n+1) :
    if i%2==0 :
        print(i)
```

À propos de range : Grâce à `range`, il est possible de considérer des entiers en progression arithmétique compris entre deux entiers. La syntaxe est alors la suivante : `range(debut,fin,raison)`. L'entier `debut` est considéré. Par contre, l'entier `fin`, lui ne l'est pas.

Exemple : afficher les nombres pairs compris entre 1 et n

```
for i in range(2,n+1,2) :
    print(i)
```

Exemple : afficher les entiers de 1 à n dans l'ordre décroissant

```
for i in range(n,0,-1) :
    print(i)
```