For

*#Quand on utilise la boucle for, on peut parcourir une collection par élément de la collection. Par exemple, si j'ai une liste < l = [4, 6, 8, 16] > et que je fais < for n in l: >, ça va créer une boucle qui va parcourir les 4 chiffres, une boucle qui passe 4 fois. < m = test, for c in m > va venir faire passer successivement "t" "e" "s" "t" car les éléments de ma collection sont des lettres. Deuxième chose, si on a un ensemble de valeurs comme le <l> du dessus et que je veux tester qu'une valeur se trouve dedans, j'écris < 4 in l > (sans for !), dans ce cas-là, ça permet de voir si 4 est à l'intérieur de ce groupe-là mais si j'avais mis < 5 in l >, ça aurait été faux. Ça vaut pour les lettres.*

On peut utiliser autre chose qu’un len() ou range() dans une boucle for. Une chaine de caractères dans un mot peut être comparée sans devoir être mise dans une liste sauf si on veut utiliser un méthode comme « .append, .pop, .insert, .remove » qui ne peuvent être utilisée qu’avec une vraie liste/un tableau (array).

In

On peut écrire : not in

Exemple : *if* v not in vowels :

con\_list.append(v)

Exemple 2 :

*If* v not (a-b)

Fonctions

# Return

La fonction RETURN arrête la boucle et renvoie la(une seule) valeur demandée.