----------------------------------------------------------

NOTIONS AVANCEES : FONCTIONS RECURSIVES  
----------------------------------------------------------

Exemple 1 -> fixer une limite pour éviter de boucler a l'infini :  
def a(n, limit): # Fonction normale  
 if n > limit:  
 return # on arrête la récursion avant de dépasser la limite.  
 # return permet d'arrêter la récursion et de casser la boucle infinie.  
 print("n:", n)  
 a(n\*n, limit)  
 # A chaque fois que l'on appelle la fonction, on multiplie le nbre par lui-même.  
 # On s'arrête lorsque la valeur atteint une certaine limite.  
  
  
# Appel de la fonction a.  
a(2, 100000) # 2 représente ici "n" et 100000 est la limite.

*"""  
Cette fonction print ... puis s'appelle elle-même, donc on re-rentre dans a, on rappelle ..."  
Cela crée une boucle infinie.  
Les fonctions récursives doivent tjs avoir une condition de sortie.  
Elles sont souvent utilisées pour le traitement des données, algorithmes, fonctions maths.  
"""*Exemple 2 -> Demander à l'utilisateur un nombre entre 1 et 4 :  
def demander\_choix\_utilisateur(min, max):  
 # On retourne ce choix au format chaine de caractères :  
 reponse\_str = input("Quel est votre choix entre " + str(min) + " et " + str(max) + " ? ")  
 # Cette conversion peut échouer donc on ajoute :  
 try:  
 reponse\_int = int(reponse\_str) # Conversion au format numérique  
 if not min <= reponse\_int <= max: # Si la réponse n'est pas entre min et max  
 print("ERREUR : Vous devez rentrer un nombre entre", min, "et", max, "!")  
 return demander\_choix\_utilisateur(min, max) # Fonction récursive  
 return reponse\_int  
 except:  
 print("ERREUR : Vous devez rentrer un nombre.")  
 return demander\_choix\_utilisateur(min, max)  
 # Ici, on ne sort pas tant que l'utilisateur n'a pas donné une bonne réponse.  
 # Elle est là la fonction récursive !  
  
  
Appel de la fonction à l'utilisateur, choix entre 1 et 4 :  
choix = demander\_choix\_utilisateur(1, 4)  
print("Choix de l'utilisateur : ", choix)  
  
Ici la récursion va servir à forcer l'utilisateur à donner une bonne réponse donc reposer la question.  
  
---------------------------------------------------------  
# DIFFERENCE ENTRE BREAK ET RETURN :

---------------------------------------------------------  
  
def a():  
 print("Début de la fonction.")  
 for i in range(0, 100):  
 if i > 20:  
 break  
 print(i)  
 print("Fin de la fonction.")  
  
  
a()  
  
# Break : arrête l'exe d'une boucle. Boucle cassée avant d'atteindre la fin.  
# Return : même chose mais ici on sort de la fonction. La suite de la boucle n'est pas exécutée.  
  
---------------------------------------------------------  
CALLBACKS ET LAMBDAS :

---------------------------------------------------------  
def ma\_fonction():  
 print("Toto")  
 return 1  
  
a = 5 # Une variable prend une valeur  
b = ma\_fonction  
  
print("a", a, "b", b())  
  
# Callbacks : quand on fait des rappels de fonctions, on met des ().  
# Stocker une fonction dans une variable ou la passer dans une autre variable = callback.  
  
Exemple :  
  
# Fonction pour multiplier :  
def mult\_callback(a, b):  
 return a\*b  
  
# Fonction pour additionner :  
def add\_callback(a, b):  
 return a+b  
  
# Fonction generique en utilisant les callbacks :  
def afficher\_table\_(n, operateur\_str, operation\_cbk):  
# On ajoute un operateur au format chaine de caractere.  
# operateur\_cbk -> fonction qui exécute l'opération  
 for i in range(1, 10):  
 print(i, operateur\_str, n, "=", operation\_cbk(i, n))  
  
  
# Fonctions lambda -> fonctions qui n'ont pas de nom. On inscrit le code directement :  
# C'est le cas ici pour mult\_callback par exemple.  
afficher\_table\_(2, "x", mult\_callback)  
-> afficher\_table\_(2, "x", lambda a, b : a \* b)  
print()  
afficher\_table\_(2, "+", add\_callback)

--------------------------------------------------------

NOMBRE VARIABLE DE PARAMETRES :  
--------------------------------------------------------  
def a(age, taille, numero\_tel):  
 print("Toto", age, taille, numero\_tel)  
  
a(20, 1.75, "0610101010")  
  
Exemple avec \* :  
def somme(titre, \*nombres):  
 print("TITRE:", titre)  
 resultat = 0  
 for n in nombres:  
 resultat += n  
 return resultat  
  
print(somme("somme des points", 5, 2, 4, 7, 8))  
  
Exemple avec \*\* :  
def somme(titre, \*\*nombres):  
 print("TITRE:", titre)  
 resultat = 0  
 for n in nombres.values():  
 resultat += n  
 return resultat  
  
print(somme("somme des notes", maths=15, geo=11, anglais=18))  
# nombres.values() -> On récupère pas de nombres mais des valeurs