➢ Notion de fonction

En Python, avant le programme principal, on peut définir des fonctions qui seront utiles dans celui-ci.
Ces fonctions peuvent utiliser des paramètres ou non, renvoyer une valeur ou non.
On peut même les inclure dans un autre programme et les appeler avec un *import* (comme *turtle* ou *random*)

➢ Structure et définition d'une fonction

def fonction(m, n, …) :	Définition du nom de la fonction et des paramètres qu'elle va utiliser.
global a, b	Permet la modification de variables du programme principal.
…	Corps de la fonction qui a la même syntaxe qu'un programme principal.
return v	Permet à la fonction de renvoyer une valeur.

Important :
Ne pas oublier les « : » en bout de ligne et l'indentation du bloc d'instructions…

Paramètres :
m, n, … sont optionnels on peut écrire : *def* fonction() :
On peut attribuer une valeur par défaut à ces paramètres avec un « = ».
Les variables utilisées dans le corps de la fonction ne modifient pas celles utilisées dans le programme principal même si elles ont le même nom ! On peut lire sans problème les valeurs des variables du programme principal, mais si on veut les modifier il faut alors utiliser l'instruction optionnelle *global*.

Différence entre fonction et procédure :
Une fonction peut simplement effectuer une liste d'instruction (elle est alors appelée procédure), mais elle peut aussi renvoyer une valeur, il faut alors utiliser l'instruction optionnelle *return*.

Récursivité :
Une fonction peut s'appeler elle-même, mais attention à bien vérifier qu'elle est bien définie et qu'elle ne va pas tourner sans jamais s'arrêter !

➢ Exemple

Soit u la suite géométrique de premier terme $u_0 = 2$ et de raison $q = 3$.

Mode explicite : $u_n = u_0 \times q^n$

```
def u(n) :
    v = 2 * (3**n)
    return v
```

Mode récurrent : $u_{n+1} = q \times u_n$

```
def u(n) :
    if n == 0 :
        return 2
    else :
        v = 3 * u(n – 1)
        return v
```