# Les Fonctions:

- ✓ En informatique, une **fonction** est une suite d'instructions qui seront exécutées à l'appel de la fonction .

- ✓ Les fonctions permettent de découper le problème global en éléments plus simples.

- ✓ En effet, lorsqu'une tâche doit être réalisée plusieurs fois par un programme avec seulement des paramètres différents, on peut l'isoler au sein d'une fonction :

❖ Exemple les fonctions prédéfinies : print(), input(), range(), len().

```
>>> print('Les fontions ça facilite la programmation')
Les fontions ça facilite la programmation
>>>
```

- ✓ Une **fonction** peut comporter des ***paramètres*** d'entrée que l'on nomme ***arguments***. Travaillant sur ces arguments, elle retourne généralement une **valeur de sortie**.

# Déclaration d'une fonction:

✓ En Python une fonction est nommée **définition**. La syntaxe pour définir une fonction est la suivante:

```
def nom_de_la_fonction(paramètres1,paramètres2,etc):
    bloc d_instructions
```

✓ Lorsque la fonction est appelée, le bloc d'instruction est exécuté, et un résultat peut-être retourné à l'appelant grâce à l'instruction ***return*** **(*valeur*** **)**

À tester

```
>>> def carre(n):
        p=n*n
        return(p)

>>> carre(5)
25
>>> 2*carre(6)
72
>>> var=carre(4)
>>> var
16
```

Fonction qui calcule le carré de n

Retour des résultats

Retour du résultat dans un variable

# Exemples d'utilisation des fonctions:

✓ Une fonction qui fait appel à une autre fonction:

À tester

```
>>> def carre(n):
        p=n*n
        return(p)

>>> def cube(m):
        c=carre(m)*m
        return(c)

>>> cube(2)
8
>>> carre(2)*cube(3)
108
>>>
```

✓ **Remarque:** Lorsqu'une instruction ***return (valeur )*** est exécutée dans le bloc, la valeur est renvoyée et **l'exécution de la fonction s'arrête.**

À tester

```
>>> def carre(n):
        p=n*n
        print('affichage avant le return')
        return(p)
        print('affichage après le return')

>>> carre(6)
affichage avant le return
36
>>>
```

# Exemples d'utilisation des fonctions:

✓ Appel d'une fonction dans un script enregistré dans un fichier (***carre.py***):

À tester

```
#fonction carré
def carre(n):
    p=n*n
    return(p)

#programme principal
nombre=float(input('entrez un nombre :  '))

#appel de la fonction
resultat=carre(nombre)

#affichage du résultat
print('le carré de ',nombre,'est :',resultat)
```

✓ **Remarque:** dans un programme, les fonctions sont écrites au début du script.

# Importer des fonctions:

✓ **Enregistrez le script suivant dans un fichier *cube.py***

```
#importer la fonction du script carre.py
from carre import *

def cube(m):
    c=carre(m)*m
    return(c)

#programme principal
nombre=float(input('entrez un nombre :  '))

#appel de la fonction
resultat=cube(nombre)

#affichage du résultat
print('le cube de ',nombre,'est :',resultat)
```

✓ **Modifiez le script *carre.py* comme indiqué ci-dessous:**

```
#fonction carré
def carre(n):
    p=n*n
    return(p)

if __name__=='__main__':
    #programme principal
    nombre=float(input('entrez un nombre :  '))

    #appel de la fonction
    resultat=carre(nombre)

    #affichage du résultat
    print('le carré de ',nombre,'est :',resultat)
```

✓ **Lancer le programme *cube.py***

# Variables locales :

✓Lorsque nous définissons des variables à l'intérieur du corps d'une fonction, ces variables ne sont accessibles qu'à la fonction elle-même.

✓On dit que ces variables sont des variables locales à la fonction.

```
>>> def carre(n):
        p=n*n
        print('p=',p)
        return(p)

>>> carre(4)
p= 16
16
>>> p
Traceback (most recent call last):
  File "<pyshell#41>", line 1, in <module>
    p
NameError: name 'p' is not defined
```

✓ la variable p est déclarée dans l'espace réservé à la fonction « carre »

✓ la variable p est inconnue du programme principal!

✓ **L'espace de noms qui contient la variable p est strictement réservé au fonctionnement interne de carre(). Il est automatiquement détruit dès que la fonction a terminé son travail.**

# Variables globales :

✓ Une **variable globale** est une variable déclarée à l'extérieur du corps de toute fonction , et pouvant donc être utilisée n'importe où dans le programme.

✓ Pour rendre accessible une variable définie à l'intérieur d'une fonction à partir du programme principal, il faudra utiliser l'instruction **global.**

**Sans l'instruction global**

```
>>> def carre(n):
        p=n*n
        print('p=',p)
        return(p)

>>> carre(4)
p= 16
16
>>> p
Traceback (most recent call last):
  File "<pyshell#41>", line 1, in <module>
    p
NameError: name 'p' is not defined
```

**Avec l'instruction global**

```
>>> def carre(n):
        global p
        p=n*n
        print('p=',p)
        return(p)

>>> carre(4)
p= 16
16
>>> p
16
>>>
```

# Exercices:

- ✓ Ecrire une fonction ***somme*** qui prend en paramètres 2 entiers m et n et qui retourne la somme des entiers de m à n (on supposera que m < n).
  Par exemple, somme(5,8) donne 26 (= 5 + 6 + 7 + 8).

❖ Ecrire une fonction ***conversion_temps(h, m, s)*** qui prend en paramètre un horaire écrit sous forme (heures ,minutes, secondes) et retourne cet horaire converti en secondes.

❖En utilisant cette fonction saisir deux horaires (heures, minutes, secondes) et calculer le temps écoulé entre les deux (en secondes).

❖ Ecrire une fonction ***conversion_distance(km, m, cm)*** qui prend en paramètre une distance écrite sous forme (Km ,m, cm) et retourne cette distance en mètres.

- ❖ Ecrire une fonction ***vitesse*** qui prend en paramètre une distance sous forme (Km ,m, cm) et un temps sous forme (heures ,minutes, secondes) et qui retourne la vitesse en m/s.

**N.B**: On utilisera les fonctions ***conversion_temps(h, m, s) et conversion_distance(km, m, cm)***

# Mini projet : Convertisseur de température

- ✓ Ecrire une programme ***Convertisseur.py*** qui permet de convertir une température en Kelvin, ou en degré Celsius ou en degré Fahrenheit selon le souhait de l'utilisateur.

***Aide:***

- ✓ Ecrire une fonction ***ConversionKelvin*** qui prend en paramètre une température en Kelvin et qui la convertie en degré Celsius et en degré Fahrenheit .

- ✓ Ecrire une fonction ***ConversionCelsius*** qui prend en paramètre une température en degré Celsius et qui la convertie en Kelvin et en degré Fahrenheit .

- ✓ Ecrire une fonction ***ConversionFahrenheit*** qui prend en paramètre une température en degré Fahrenheit et qui la convertie en degré Celsius et en Kelvin .