TP ALGO 3 : LES FONCTIONS

Une **fonction**, en informatique, comporte :

- Une entrée : elle composée d'un ou plusieurs nombre(s), texte(s), que l'on appelle **arguments ou paramètres**.
- Un « corps » de fonction, qui exécute un travail sur ces arguments. Il est possible de créer une fonction sans argument.
- Une sortie, qui peut être un nombre, un texte, et qui est « renvoyée ». Une fonction peut aussi ne rien renvoyer et effectuer par exemple un affichage. On parle alors de **procédure**.

La syntaxe Python pour définir une fonction est la suivante :

```
def nom_de_la_fonction(argument1, argument2, ...):
    instructions
    return resultat1, resultat2 ...
```

La fonction s'interrompt dès qu'elle rencontre un return.

Dans le cas d'une procédure, il n'y a pas de return, et la fin de la fonction est donnée par l'indentation.

Exemples

- *Une fonction du type fonction mathématique : on a ici une fonction qui prend pour argument un nombre x et renvoie le résultat de* $3x+1$.

```
def f(x):
    return 3*x+1
```

On appelle cette fonction pour $x=4$ *simplement en écrivant* $f(4)$.
Dans la console, le résultat s'affiche directement :

```
In [2]: f(4)
Out[2]: 13
```

Si on utilise la fonction dans le corps du programme, il faut demander l'affichage, ou stocker le résultat dans une variable pour l'utiliser ultérieurement. Si on écrit seulement $f(4)$, *rien ne sera affiché, et le résultat sera « perdu ».*
Tester cela à l'aide du programme ci-contre. On obtient seulement deux affichages, la première instruction ne « fait » rien.

```
def f(x):
    return 3*x+1

f(4)

print (f(4))

a = f(4)
print (a)
```

- *Tester la fonction définie ci-dessous. Quels sont ses arguments ? Que renvoie-t-elle ?*

```
1   def carres (a,b):
2       diff = a**2 - b**2
3       somme = a**2 + b**2
4       return (diff, somme)
5
```

- La fonction ci-dessous est une procédure, sans argument.

```
def bienvenue():
    print ("bonjour")


bienvenue()
```

Exercices d'application

Exercice 1

On a défini une fonction Python par les instructions ci-contre.

```
def g(x):
    return 2*x*(x + 4)
```

1. Quel est son nom ?

2. Combien a-t-elle d'arguments (ou de paramètres)

3. Que renvoient les appels :

$g(1)$ $g¿)$ $g(4)$ $3*g(1)+4$

Exercice 2

On veut définir une fonction Python nommée aire qui renvoie l'aire d'un triangle de base b et de hauteur h.

1. Combien a-t-elle d'arguments et lesquels ?

2. Quel résultat doit-elle renvoyer ?

3. Ecrire le script définissant la fonction aire. Le tester pour une base de 5 cm et une hauteur de 4 cm.

Exercice 3

1. Ecrire une fonction qui prend en paramètres deux nombres et renvoie le plus grand des deux.

2. Même question avec trois nombres.

Exercice 4

Ecrire une fonction distance qui prend en paramètre deux nombres et renvoie la distance entre ces deux nombres.

Exercice 5

1. Ecrire une fonction triangle(n) qui trace un triangle équilatéral de côté n tel que ci-dessous :

2. Ecrire une fonction carre(n) qui trace un carré de côté n.