# Syntaxe avancé des fonctions et modules

# Les paramètres d'une fonctions

- Il est possible de donner une valeur par défaut à un paramètre d'une fonction.
- Un paramètre défini avec une syntaxe

```
def fct(param=valeur):
```

est appelé paramètre par mot-clé

```
def MaFonction(pays = "Maroc"):
    print("Je suis de : " + pays)

MaFonction("Egypte")
MaFonction("Espagne")
MaFonction()
```

#### Affichage:

Je suis de : Egypte

Je suis de : Espagne

Je suis de : Maroc

# Les paramètres d'une fonctions

On a la possibilité de donner des valeurs par défaut à plusieurs paramètres:

```
def MaFonction(x=0,y=0):<br/>return x+yAffichage:print( MaFonction(35,5) )<br/>print( MaFonction(20) )<br/>print( MaFonction() )20<br/>o
```

Il est aussi possible de donner le nom de l'argument lorsqu'on fait l'appel:

```
print( MaFonction(y=18) )
print( MaFonction(x=13,y=45) )
```

```
Affichage: 18 58
```

# Les paramètres d'une fonctions

 Il est possible de faire un mélange d'arguments positionnels et par mot-clé. Ainsi les arguments positionnels doivent toujours être placés avant les arguments par mot-clé :

```
def fct(a, b, x=0, y=0, z=0): return a, b, x, y, z

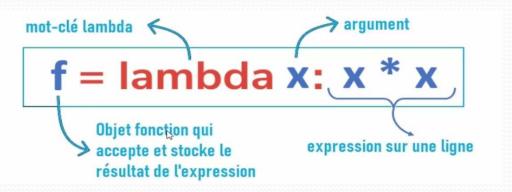
print(fct(1, 1)) #appel de la fonction avec a=b=1 et x=y=z=0 print(fct(1, 1, z=5)) #appel de la fonction avec a=b=1, x=y=0 et z=5 print(fct(1, 1, z=5, y=32))) #appel de la fonction avec a=b=1, x=0, y=32 et z=5
```

### Les fonctions lambda

• La fonction **lambda** est une petite fonction contenant **qu'une seule expression**. Elle peut agir sous anonymat parce qu'elle ne nécessite aucun nom. Elles sont très utiles lorsqu'il faut effectuer des petites tâches avec moins de code.

#### Les fonctions lambda

Syntaxe :



Pour appeler cette fonction, on écrit: print(f(5)) #25

• Équivalente à:

**def** carre(**x**): **return x**\***x** 

#### Certaines fonctions utiles

- abs(x): Permet de faire disparaitre le signe de x. Par exemple abs(3.65) vaut 3.65 et abs(-4.3) vaut 4.3
- round(x,n): Donne l'arrondi de x avec n chiffres après la virgule. Par exemple round(12.345678,3) vaut 12.345
- min(a,b) et max(a,b) : Donnent respectivement le plus petit et le plus grand des nombres entre a et b

### Notions de modules

- Il existe plusieurs fonctions prédéfinis en python, mais ces fonctions ne sont pas utiliser directement, elles se trouve dans des modules.
- Un module en langage python, est un fichier python prédéfinie qui regroupe un ensemble de fonctions prêtes à être utilisées. C'est une sorte de bibliothèque qui regroupe des fonctions.
- Pour utiliser les fonctions définie dans un module, il faut importer ce module ou importer exactement la fonction souhaitée.

#### Utiliser les modules

 Pour appeler un module dans un fichier, on utilise la syntaxe suivante : import nomModule

Puis pour travailler avec une de ses fonctions, on écrit: **nomModule.**NomFonction

2. Si on veut préciser une fonction bien déterminé dans l'importation on peut la référer directement comme suit:

from nomModule import nomFonction

ou bien:

from nomModule import \*

Étoile ici (\*) signifie **toutes** les fonctions.

## Le module mathématique: math

 Si on a un programme qui utilise des fonctions mathématiques, il faut d'abord importer le module math à l'aide de l'instruction:

#### import math

 Ce module (ou bibliothèque) est constituée de plusieurs fonctions que vous allez trouver dans le tableau dessous:

## Les fonctions du module math

Nom de la fonction	Signification	Exemple
ceil(x)	Retourne l'entier immédiatement supérieur au nombre réel x	math.ceil(7.5) -> 8
floor(x)	Retourne l'entier immédiatement inféreiur au nombre réel x	math.floor(7.5) ->7
factorial(x)	Retourne le factoriel d'un entier x	math.factorial(4) ->24
sqrt(x)	Retourne la racine carrée de x	math.sqrt(16) ->4
pow(x,y)	Retourne x puissance y	math.pow(5,2) ->25
log(x)	Retourne le logarithme népérien de x	math.log(1) ->o
log10()	Retourne le logarithme décimal de x	math.log(10) ->1
Sin(x)	Retourne le sinus de x	math.sinus(180) -> -0.8011
Cos()	Retourne le cosinus de x	math.cos(180) -> -0.5984
pi	Constante qui représente $\pi$ mathématique	math.pi

## Exemples

• Exemples d'utilisation de fonctions mathématiques:

```
#exemple 1
import math
x=math.log(1)
print(x)
#exemple 2
from math import floor
y=floor(18.123)
print(y)
```

#### Le module random

 Ce module contient des fonctions relatifs à la génération des nombre aléatoires.

#### import random

fonction	Signification	Exemple
random()	Retourne une valeur réelle aléatoire entre o et 1 exclus	X=random()
randint(x,y)	Retourne un entier entre x et y	X=radint(5,25)
choice(L)	Retourne une valeur aléatoire d'une liste L passée en paramètre.	L=[15,4,8,2,30] X=choice(L)
shuffle(L)	Permet de permuter d'une manière aléatoire les éléments d'une liste L	L=[15,4,8,2,30] Shuffle(L)