

Premiers pas en programmation

Chap. 2,1

Exercice 1. Découverte et utilisation de deux fonctions fondamentales

Les deux fonctions intégrées `dir` et `help` sont fondamentales. Les utiliser *dans la console* pour répondre aux questions ci-dessous.

1. La fonction intégrée `sum` intégrée au langage calcule-t-elle la somme de deux nombres ?
2. Que réalisent les fonctions intégrées `int`, `float`, `str`, `bool` ?
3. Quel traitement réalise la fonction `pow` du module `math` ? Donner un exemple d'utilisation.
4. Comment se nomme la fonction du module `math` qui calcule la racine carrée d'un nombre passé en argument ?

Réponse 1.

1. *Affichage de la spécification d'une fonction* : Utiliser la fonction `help`.

D'après la documentation la fonction `sum` attend une séquence de nombres et non pas deux nombres.

```
>>> help(sum) Help on built-in function sum in module builtins:
sum(iterable, start=0, /) Return the sum of a 'start' value (default: 0) plus
an iterable of numbers
When the iterable is empty, return the start value. This function is intended
specifically for use with numeric values and may reject non-numeric types.
```

2. La fonction `int` convertit n'importe quel nombre ou une chaîne de caractères en nombre entier (si possible).

```
>>> help(int) Help on class int in module builtins:
class int(object) | int([x]) -> integer | int(x, base=10) -> integer | | Convert
a number or string to an integer, or return 0 if no arguments | are given. If
x is a number, return x.__int__(). For floating point | numbers, this truncates
towards zero.
```

De même les fonctions `float`, `str` et `bool` convertissent, si possible, un nombre, une chaîne de caractères en approximation des nombres réels, chaîne de caractères, nombre booléen.

3. L'application de la fonction `pow` à deux nombres x et y donne le nombre x^y .

```
>>> import math as m
>>> help(m.pow) Help on built-in function pow in module math:
pow(x, y, /) Return x**y (x to the power of y).
```

4. *Méthode de recherche d'une fonction mathématique* : 1) Importer le module `math` ; 2) Lister, avec la fonction `dir`, toutes les fonctions de ce module et rechercher celle qui semble réaliser l'opération souhaitée ; 3) Confirmer le choix effectué à l'aide de la fonction `help`.

```
>>> import math
>>> dir(math) ['__doc__', '__file__', '__loader__', '__name__', '__package__',
 '__spec__', 'acos', 'acosh', 'asin', 'asinh', 'atan', 'atan2', 'atanh', 'ceil',
 'copysign', 'cos', 'cosh', 'degrees', 'e', 'erf', 'erfc', 'exp', 'expm1', 'fabs',
 'factorial', 'floor', 'fmod', 'frexp', 'fsum', 'gamma', 'gcd', 'hypot', 'inf',
 'isclose', 'isfinite', 'isinf', 'isnan', 'ldexp', 'lgamma', 'log', 'log10',
 'log1p', 'log2', 'modf', 'nan', 'pi', 'pow', 'radians', 'remainder', 'sin',
 'sinh', 'sqrt', 'tan', 'tanh', 'tau', 'trunc']
>>> help(math.sqrt)
Help on built-in function sqrt in module math:
sqrt(x, /) Return the square root of x.
```

Exercice 2. Utilisation d'une fonction

1. Peut-on appeler la fonction `randint` du module `random` sans argument ?

Rappel. La fonction intégrée `help` permet d'obtenir la documentation de n'importe quelle fonction.

2. Que retourne l'appel de la fonction `randint` du module `random` ?
3. Appeler cette fonction `randint` de façon à obtenir un nombre entier aléatoire compris entre 1 et 10.
4. Est-il possible que l'appel précédent retourne 1 ou 10 ?

Réponse 2.

```
>>> import random
>>> help(random.randint) Help on method randint in module random:
randint(a, b) method of random.Random instance Return random integer in range [a, b],
including both end points.
```

1. La documentation nous apprend que la fonction attend deux arguments.
2. La fonction retourne un entier compris entre les deux bornes passées en argument.
3.

```
>>> random.randint(1, 10)
```
4. La fonction retourne un entier compris entre 1 et 10 inclus.

Exercice 3. Définition d'une fonction

1. La fonction définie ci-dessous est-elle syntaxiquement correcte ?

```
def retire_un(n):
    return 1 - n
```

2. La spécification est la suivante :

```
def retire_un(n: float) -> float:
    """ Retourne la valeur de n - 1 """
```

La fonction telle qu'elle est définie à la question 1 correspond-elle à cette spécification ?

3. Corriger le corps de la fonction `retire_un` de façon à ce que le calcul effectué corresponde à ce qu'annonce la spécification.

Réponse 3.

1. Du point de vue syntaxique la fonction est correcte.
2. La fonction telle qu'elle est définie retourne la valeur de $1 - n$ et ne correspond donc pas à la spécification.
- 3.

```
def retire_un(n: float) -> float:
    """ Retourne la valeur de n - 1 """
    return n - 1
```

Exercice 4. Calcul de fonctions polynomiales

1. Définir et tester une fonction écrite en Python, nommée `polynomiale`, telle que `polynomiale(a, b, c, d, x)` retourne la valeur de la fonction qui à x associe $ax^3 + bx^2 + cx + d$.

La documentation complète de cette fonction est la suivante :

```
def polynomiale(a:int, b:int, c:int, d:int, x: float) -> float:
    """ Retourne la valeur ax^3 + bx^2 + cx + d
    >>> polynomiale(1, 1, 1, 1, 2) 15.0
    >>> polynomiale(1, 1, 1, 1, 3) 40.0
    """
```

2. Définir et tester une fonction écrite en Python, nommée `polynomiale_carre` qui retourne la valeur de la fonction $ax^4 + bx^2 + c$.

La documentation complète de cette fonction est la suivante :

```
def polynomiale_carre(a:int, b:int, c:int, x: float) -> float:
    """ Retourne la valeur ax^4 + bx^2 + c
    >>> polynomiale(1, 1, 1, 1, 2) 15.0
    >>> polynomiale(1, 1, 1, 1, 3) 40.0
    """
```

Réponse 4.

- 1.

```
from math import pow
def polynomiale(a:int, b:int, c:int, d:int, x: float) -> float:
    """ Retourne la valeur ax^3 + bx^2 + cx + d
    >>> polynomiale(1, 1, 1, 1, 2) 15.0
    >>> polynomiale(1, 1, 1, 1, 3) 40.0
    """
    return a * pow(x, 3) + b * pow(x, 2) + c * x + d
```

2.

```
import math
def polynomiale_carre(a:int, b:int, c:int, x: float) -> float:
    """ Retourne la valeur  $ax^4 + bx^2 + c$ 
    >>> polynomiale(1, 1, 1, 1, 2) 15.0
    >>> polynomiale(1, 1, 1, 1, 3) 40.0
    """
    return a * pow(x, 4) + b * pow(x, 2) + c
```

Exercice 5. Calcul de la moyenne de trois nombres

1. Définir et tester la fonction nommée `somme`, écrite en Python, dont la spécification est :

```
def somme(x:float, y: float, z: float) -> float:
    """ Retourne la somme des trois nombres passés en argument. """
```

2. Définir et tester une fonction écrite en Python, nommée `moyenne`, qui détermine la moyenne arithmétique de trois nombres.

Remarque. La fonction `moyenne` doit utiliser la fonction `somme` définie à la questions précédente pour effectuer son traitement.

La spécification de la fonction est la suivante :

```
def moyenne(a: float, b: float, c: float) -> float:
    """ Retourne la moyenne des trois nombres passés comme arguments. """
```

3. Définir et tester une fonction écrite en Python, nommée `moyenne_ponderee`, qui détermine la moyenne pondérée de trois nombres avec des coefficients variables. Les paramètres devront être écrits dans l'ordre suivant : *d'abord les trois nombres, puis les trois coefficients*.

Remarque. La fonction `moyenne_ponderee` doit effectuer deux appels à la fonction `somme` pour effectuer son traitement.

La documentation complète de la fonction est la suivante :

```
def moyenne_ponderee(x:float, y:float, z:float, a:int, b:int, c:int) -> float:
    """ Retourne la moyenne pondérée des nombres x, y et z par les coefficients a,
    b et c.

    >>> moyenne_ponderee(5, 10, 15, 1, 1, 1)
    10.0

    >>> moyenne_ponderee(5, 10, 15, 1, 1, 0)
    7.5
    """
```

Réponse 5.

```
def somme(x:float, y: float, z: float) -> float:
    """ Retourne la somme des trois nombres passés en argument. """
    return x + y + z

def moyenne(a: float, b: float, c: float) -> float:
    """ Retourne la moyenne des trois nombres passés comme arguments. """
    return somme(a, b, c) / 3

def moyenne_ponderee(x:float, y:float, z:float, a:int, b:int, c:int) -> float:
    """ Retourne la moyenne pondérée des nombres x, y et z par les coefficients
    a, b et c.

    >>> moyenne_ponderee(5, 10, 15, 1, 1, 1)
    10.0
    >>> moyenne_ponderee(5, 10, 15, 1, 1, 0)
    7.5
    """
    return somme(a * x, b * y, c * z) / somme(a, b, c)
```

Exercice 6. Calculs de surfaces et de volumes

1. Définir et tester une fonction écrite en Python, nommée `surface_rectangle`, qui détermine la surface d'un rectangle de longueur a et de largeur b .

Préciser la spécification de la fonction.

2. Définir et tester une fonction écrite en Python, nommée `volume_parallelepipede`, qui détermine le volume d'un parallélépipède rectangle de longueur a , de largeur b et de hauteur h .

Préciser la spécification de la fonction.

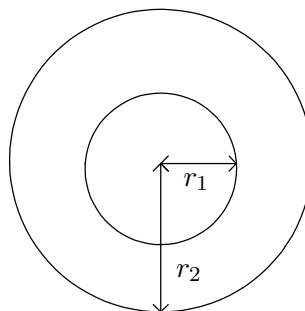
Cette fonction devra utiliser la fonction `surface_rectangle` pour effectuer son traitement son traitement.

3. Définir et tester une fonction écrite en Python, nommée `surface_disque`, qui détermine la surface d'un disque de rayon r .

Rappel. Le module `math` possède une variable `pi`.

Préciser la spécification de la fonction.

4. Définir et tester une fonction écrite en Python, nommée `surface_couronne`, qui détermine la surface d'une couronne de rayon intérieur r_1 et de rayon extérieur r_2 .



Préciser la spécification de la fonction.

Cette fonction devra utiliser la fonction `surface_disque` pour effectuer son traitement.

5. Définir et tester une fonction écrite en Python, nommée `volume_tube`, qui détermine le volume de la partie pleine d'un tube de longueur l , dont la section est une couronne de rayon intérieur r_1 et de rayon extérieur r_2 .

Préciser la spécification de la fonction.

Cette fonction devra utiliser la fonction `surface_couronne` pour effectuer son traitement.

Réponse 6.

```
from math import pow, pi

def surface_rectangle(a: float, b: float) -> float:
    """ Détermine la surface du rectangle de côtés a et b. """
    return a * b

def volume_parallelepipede(a: float, b: float, h: float) -> float:
    """ Retourne le volume du parallélépipède rectangle de côtés a, b
    et de hauteur h. """
    return surface_rectangle(a, b) * h

def surface_disque(r: float) -> float:
    """ Retourne la surface du disque de rayon r. """
    return pi * pow(r, 2)

def surface_couronne(r1: float, r2: float) -> float:
    """ Retourne la surface de la couronne comprise entre les rayons
    r1 et r2. """
    return surface_disque(r2) - surface_disque(r1)

def volume_tube(r1, r2, l) -> float:
    """ Retourne le volume de la partie pleine d'un tube de longueur l,
    dont la section est une couronne de rayon intérieur r1 et de rayon
    extérieur r2. """
    return surface_couronne(r1, r2) * l
```

Exercice 7. Calcul d'une distance

Définir et tester une fonction écrite en Python, nommée `distance` qui détermine la distance séparant deux points de coordonnées (x_1, y_1) et (x_2, y_2) d'un plan.

La documentation complète de la fonction est la suivante :

```
def distance(x1: float, y1: float, x2: float, y2: float) -> float:
    """ Retourne la distance dans le plan entre les deux points de coordonnées
    (x1, y1) et (x2, y2).

    >>> distance(0, 0, 1, 1)
    1.4142135623730951
```

"""

Réponse 7.

```
from math import pow, sqrt
def distance(x1: float, y1: float, x2: float, y2: float) -> float:
    """ Retourne la distance dans le plan entre les deux points de coordonnées
        (x1, y1) et (x2, y2).

    >>> distance(0, 0, 1, 1)
    1.4142135623730951
    """
    return sqrt(pow(x2 - x1, 2) + pow(y2 - y1, 2))
```

Exercice 8. Reprise de l'exercice 1

L'objectif de cet exercice est de retrouver une partie du comportement de la fonction `randint` du module `random` à partir de la fonction `random` de ce même module.

Définir et tester une fonction écrite en Python, nommée `tirage_entier`, telle que `tirage_entier(a)` retourne un nombre entier compris entre 1 et a (inclus), au hasard. *La fonction `tirage_entier` doit utiliser la fonction `random` du module `random`.*

Remarque. Lire attentivement la documentation de la fonction `random` afin de comprendre ce qu'elle retourne.

Rappel. La fonction intégrée `int` permet de convertir n'importe quel nombre en nombre entier.

Réponse 8.

```
from random import random
def tirage_entier(a: int) -> int:
    """ Retourne un nombre entier aléatoire compris entre 1 et a (inclus). """
    return int(random() * a + 1)
```