

Les fonctions

1 Généralités

Définition

Une fonction est un **sous-programme** qui regroupe plusieurs instructions. Une fonction peut recevoir des données en entrée (appelées **arguments**) et/ou **retourner** des données en sortie grâce au mot-clef **return**. On lui donne un **nom** qui servira à l'utiliser en l'appelant à chaque fois qu'on le souhaite.

En Python :

- La déclaration d'une fonction se fait à l'aide du mot-clef `def` ;
- Les instructions du corps de la fonction doivent être indentées ;
- Le nom d'une fonction commence par une lettre minuscule ;
- On fera attention de ne pas utiliser un mot réservé par Python pour nommer sa fonction !

Exemple : `print` est déjà le nom d'une fonction dans la bibliothèque standard Python.

Exemple : On écrit la définition d'une fonction qui permet de tracer un angle droit à l'aide du module Turtle :

```
1 from turtle import forward, left
2
3 def trace_angle_droit():
4     forward(100)
5     left(90)
6     forward(100)
```

Attention

Ici, nous avons défini notre fonction, c'est-à-dire que si on exécute le fichier la contenant tel quel, il ne se passera rien du tout. Pour voir ce que fait la fonction, il est fondamental de l'appeler !

```
1 from turtle import forward, left
2
3 def trace_angle_droit():
4     forward(100)
5     left(90)
6     forward(100)
7
8 trace_angle_droit() # appel de la fonction
```

Les fonctions sont **documentées** à l'aide de *docstrings* : il s'agit d'une description située entre triple guillemets doubles droits (`"""`) juste après l'entête de la fonction. Il est possible d'accéder à cette documentation grâce à la fonction `help` fournie par la bibliothèque standard de Python. Vous pouvez déjà l'essayer dans une console Python en tapant `help(print)` pour commencer.

```
>>> help(print)
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
```

Optional keyword arguments:
file: a file-like object (stream); defaults to the current sys.stdout.
sep: string inserted between values, default a space.
end: string appended after the last value, default a newline.
flush: whether to forcibly flush the stream.

Reprenons notre **exemple** précédent et ajoutons une docstring :

```
1 from turtle import forward, left
2
3 def trace_angle_droit():
4     """Trace un angle droit"""
5     forward(100)
6     left(90)
7     forward(100)
```

Puis, dans notre console :

```
>>> help(trace_angle_droit)
Help on function trace_angle_droit in module __main__:

trace_angle_droit()
    Trace un angle droit
```

C'est une **bonne pratique** de prendre l'habitude de bien documenter ses programmes en utilisant notamment les docstrings. Il est conseillé d'y préciser les paramètres d'entrée et la valeur renvoyée.

Exercice 1

Que fait le programme suivant ? Expliquer pourquoi en citant les numéros de ligne pour justifier.

```
1 from turtle import forward, left
2
3 def trace_angle_droit():
4     forward(100)
5     left(90)
6 forward(100)
7
8 trace_angle_droit() # appel de la fonction
```

2 Passage de paramètres dans une fonction

En Python, lors de l'appel d'une fonction qui prend un ou plusieurs arguments, tout se passe **en général** comme si le programme fournissait à la fonction une copie de l'argument : changer un paramètre dans une fonction n'affecte donc pas sa valeur à l'extérieur de la fonction (voir cas particulier ci-après). On donne deux exemples ci-dessous.

```
1 def foo(u):
2     u = u+1
3     print(u) # affiche 2
4
5 x = 1
6 foo(x)
7 print(x) # affiche 1
```

```
1 def foo(u):
2     x = x+1
3     print(x) # affiche 2
4
5 x = 1
6 foo(x)
7 print(x) # affiche 1
```

Exercice 2

1. Reprendre le programme en exemple du 1 (`trace_angle_droit`) et faire en sorte que la fonction prenne un paramètre `cote` de manière à ce qu'on puisse choisir la longueur des côtés formant l'angle droit.
2. Écrire la documentation de cette fonction en faisant attention de respecter les bonnes pratiques.

3 Renvoi d'une valeur

En début d'année, avant de connaître l'instruction `return`, les fonctions considérées étaient en réalité des procédures : leur corps est constitué d'une séquence d'instructions que le programme exécute, et la dernière ligne en est l'unique point de sortie.

C'est le cas de l'exemple suivant :

```
1 def foo(n):
2     if n%2==0 :
3         print(n, "est pair")
4     else :
5         print(n, "est impair") # point de sortie
```

Mais la plupart des langages évolués permettent à une fonction de renvoyer une valeur, qui pourra être exploitée dans une autre partie du code. En Python, le renvoi d'une valeur s'effectue avec le mot-clef `return`.

Exercice 3

À faire d'abord sur papier puis à vérifier sur ordinateur : quels sont les affichages produits par le script ci-dessous ?

```
1 def foo(n):
2     if n%2==0 :
3         m = n/2
4     else :
5         m = 3*n+1
6     return m
7
8 a = 5
9 a = foo(a)
10 print(a) # affiche ...
11 foo(a)
12 print(a) # affiche ...
```

Propriété

Les deux définitions suivantes de la fonction `foo` sont donc équivalentes.

```
1 def foo(n):
2     if n%2==0 :
3         m = n/2
4     else :
5         m = 3*n+1
6     return m
```

```
1 def foo(n):
2     if n%2==0 :
3         m = n/2
4         return m # point de sortie
5     m = 3*n+1
6     return m # point de sortie
```

Exercice 4

Avec la définition suivante, combien de tours de la boucle `while` sont effectués à l'appel `est_premier(55)` ?

```
1 def est_premier(n):
2     d = 2
3     while d < n :
4         if n % d == 0 :
5             return False
6         d = d + 1
7     return True
```

Exercice 5

Écrire une fonction `aire_trapeze(a, b, h)` qui renvoie l'aire d'un trapèze de bases a et b et de hauteur h .

→ L'aire d'un trapèze de bases a et b et de hauteur h est donnée par la formule $\frac{(a+b)h}{2}$.

Exercice 6

Écrire une fonction `aire_triangle(a, b, c)` qui renvoie l'aire \mathcal{A} d'un triangle de côtés a , b et c .

→ On pourra utiliser la formule de Héron :

$$\mathcal{A} = \sqrt{p(p-a)(p-b)(p-c)} \quad \text{avec} \quad p = \frac{a+b+c}{2}.$$

Exercice 7

Considérons une fonction `est_rectangle(xa, ya, xb, yb, xc, yc)` qui, étant données les coordonnées de trois points A, B, C du plan, affiche `True` si le triangle ABC est rectangle, et `False` sinon. Une telle fonction va nécessairement contenir du code redondant, par exemple le calcul de AB^2 , AC^2 , et BC^2 si l'on utilise le théorème de Pythagore.

```
1 def est_rectangle(xa, ya, xb, yb, xc, yc):
2     ab2 = (xb-xa)**2+(yb-ya)**2
3     ac2 = (xc-xa)**2+(yc-ya)**2
4     bc2 = (xc-xb)**2+(yc-yb)**2
```

Il devient alors intéressant d'utiliser une fonction auxiliaire `distance_au_carre` qui va contenir le code responsable du calcul des carrés des distances.

```
1 def distance_au_carre(x1, y1, x2, y2):
2     return (x2-x1)**2+(y2-y1)**2
3
4 def est_rectangle(xa, ya, xb, yb, xc, yc):
5     ab2 = distance_au_carre(xa, ya, xb, yb)
6     ac2 = distance_au_carre(xa, ya, xc, yc)
7     bc2 = distance_au_carre(xa, ya, xc, yc)
```

1. Compléter la fonction `est_rectangle` pour qu'elle renvoie `True` si ABC est rectangle, et `False` sinon.
2. On considère les points $A(5; 2)$, $B(2; 2)$ et $C(2; 6)$.
Vérifier que l'appel `est_rectangle(5, 2, 2, 2, 2, 6)` renvoie bien la valeur `True` dans ce cas.
3. Proposer une autre solution au problème s'appuyant sur le produit scalaire.

Exercice 8

Soit f la fonction définie par $f(n) = n^2 - n + 41$ pour tout entier naturel n .

1. Vérifier que si $0 \leq n \leq 40$, alors $f(n)$ est un nombre premier.
2. Parmi les entiers naturels $n \leq 100$, combien d'entre eux sont tels que $f(n)$ n'est pas un nombre premier ?

Exercice 9

Pour tout entier naturel $n > 0$, on considère les sommes

$$s_1(n) = 1 + 2 + 3 + \cdots + n \quad \text{et} \quad s_3(n) = 1^3 + 2^3 + 3^3 + \cdots + n^3.$$

1. Vérifier que $s_1(100) = 5\,050$ et que $s_3(100) = 5\,050^2$.
2. Existe-t-il une valeur de n inférieure à 1000 telle que $s_3(n) \neq s_1(n)^2$?

La valeur None

Dans une fonction En Python, si aucun renvoi de valeur n'est spécifié (absence de `return` dans le corps de la fonction), alors celle-ci renvoie l'objet spécial `None`.