

Les fonctions en python



Une **fonction** est un sous-programme que l'on peut appeler et exécuter plusieurs fois.

```
from math import sqrt

def hypotenuse(a, b) :
    # Théorème de Pythagore
    c = sqrt(a ** 2 + b ** 2)
    # La fonction renvoie le résultat
    return c

def reponse(n):
    return n == 42
```

```
>>> hypotenuse(3, 4)
# renvoie
>>> reponse(4)
# renvoie
```

Sur la **NumWorks**, la touche **boîte à outils**  permet d'accéder aux instructions, et la touche **var**  permet d'appeler les fonctions disponibles.

Instructions conditionnelles

Une instruction conditionnelle n'est exécutée que si une condition est réalisée.

```
def ranger(a, b) :
    if a < b :
        return a, b
    else:
        return b, a
```

Les structures conditionnelles, les boucles, les fonctions finissent leur déclaration initiale par ":" et sont suivies de leurs instructions qui seront indentés (décalés de 2 ou 4 espaces).

Les boucles bornées for

Permet de répéter un nombre fixé de fois un bloc d'instructions indenté.

```
# Affiche la table de 7
nb = 7
for i in range(11):
    print(nb, "x", i, "=", nb * i)
```

Les boucles non bornées while

Permet de répéter plusieurs fois un bloc d'instructions indenté jusqu'à ce qu'une condition d'arrêt se produise.

```
# sapin.py
hauteur = 6
diese = 1
espace = hauteur - diese

while espace > 0 :
    print(" " * espace + "#" * diese )
    espace -= 1
    diese += 2

print(" " * (hauteur - 1) + "#")
```

```
deg PYTHON
>>> from sapin import *
#
###
#####
#####
#####
#####
#####
#####
#####
#
#
>>> |
```

remix de nsi.xyz/start-p3
cc by sa by Vincent ROBERT



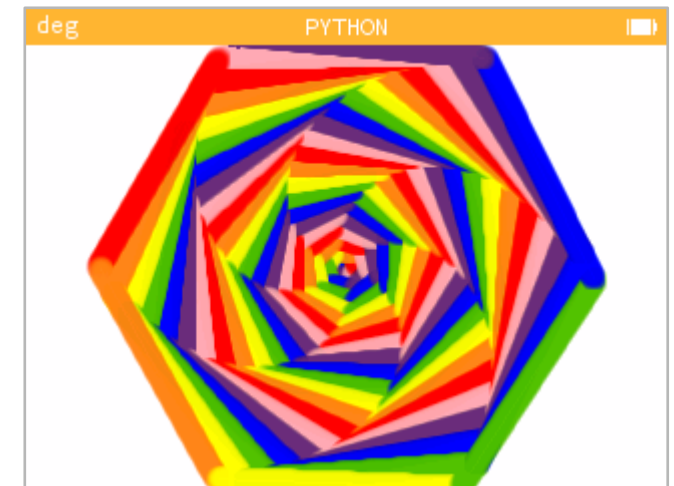
Débuter et découvrir python3

Python est un langage de programmation simple d'usage qui sera utilisé en mathématiques ainsi qu'en spécialité **NSI (Numérique et Sciences Informatiques)** lors de diverses activités.

```
from turtle import *
liste = ["pink", "red", "orange", "yellow", "green", "blue", "purple"]
for i in range(3*42) :
    color(liste[i%7])
    pensize(10+2)
    forward(i)
    left(59)
```

↑ un exemple de script en python

Le résultat sur une **NumWorks** ↓



Exécuter depuis une console

Sur la **NumWorks**, se rendre dans l'application python puis dans la **console d'exécution**.

Types de valeurs

```
>>> type(4)
#
>>> type(4.0)
#
>>> type(3E5)
#
>>> type("Bonjour")
#
```

Les opérateurs mathématiques

A tester	symbole	opération
32+10	+	addition
80-38	-	
6*7	*	
2**6	**	
355/113	/	
42//11 42//6	//	
42%11 42%6	%	
"pa"+"py"		
3*"pom"		

L'opérateur d'affectation

Une **Variable** est un nom que l'on donne à une valeur.

```
>>> a = 101
>>> b = 4
>>> e = a * b
>>> d = e + 2**8 + 6
>>> d = d + 1
```

a	b	e	d

Les opérateurs de comparaisons

A tester	symbole	comparaison
6==3**2	==	
8==2**3		
4<4	<	
4<=4	<=	
9!=9	!=	
9!=8		

Les commentaires en python

Ceci est un commentaire en python

Un commentaire commence par un dièse, il n'est pas interprété lors de l'exécution du script.

Quelques instructions natives

```
>>> len("Bonjour")
7          # Détermine la longueur

>>> max(4**3, 3**4)
#

>>> min(4**3, 3**4)
#

>>> round(355/113, 2)
#

>>> int("256")
#

>>> str(128)
#

>>> chr(42)
#

>>> ord("*")
#
```

Importer des modules en python

Certaines instructions ne sont pas natives, il faut importer des modules pour les utiliser

```
>>> from math import pi
>>> pi
#
>>> from math import sqrt
>>> sqrt(2)
#
```

Pour importer toutes les instructions d'un module :

```
>>> from math import *
```

Utiliser l'éditeur de script

Sur la **NumWorks**, se rendre dans l'application python puis dans **Ajouter un script**