

# La fonction print en Python

## Print en Python

La fonction print permet d'afficher le contenu d'un ou plusieurs objet(s) dans la console. La syntaxe de la fonction est print suivi de parenthèses : print (a, b, c). Le message sera transformé en chaîne de caractères peu importe le type des objets passés en paramètres.

Si le contenu d'un objet ne peut pas être affiché de manière lisible pour un humain, print affichera la référence de l'objet.

test.py

```
print('commentcoder.com') # commentcoder.com
print('commentcoder.com', 42, [1, 2, 3]) # commentcoder.com 42 [1,
2, 3]

def test():
    pass

print(test) # <function test at 0x142424242>
```

La fonction print permet entre autres :

- D'afficher tout ce qu'on veut en Python
- D'afficher du texte en couleur
- De modifier son formatage avec l'aide d'autres paramètres comme sep et end

Commençons sans plus tarder à explorer toutes les capacités de la fonction print en Python 3 (et Python 2 en fin d'article) !

<https://youtu.be/n2aAnQVZOFA?t=53>

## Comment utiliser la fonction print en Python ?

Pour utiliser la fonction print en Python, on lui passe des paramètres qui seront transformés en strings :

test002.py

```
# Afficher une chaîne de caractères :

print("Bienvenue sur commentcoder.com")
# Affichera "commentcoder.com" sur la sortie standard

# Afficher un nombre :
```

```
print(42)
# Affichera "42" sur la sortie standard

# Afficher le contenu d'une variable :

nombre = 42
print(nombre)
# Affichera "42" sur la sortie standard
```

Il est aussi possible d'afficher plusieurs objets Python avec la fonction print :

[test003.py](#)

```
## Afficher plusieurs objets :

print(1, 2, 3)
# Affichera "1 2 3"

# On peut aussi mixer les types d'objets :

print('École', 42)
# Affichera "École 42"

# En utilisant l'unpacking/destructuration

liste = [1, 2, 3, 4, 5]
print(liste[0], *liste[1:-1], liste[-1])

# Affichera 1 2 3 4 5
# Explications :
# liste[0] = 1 (le premier élément de la liste)
# liste[1:-1] = [2, 3, 4] (les éléments en partant du 1er non-inclus,
# jusqu'au dernier non-inclus)
# L'opérateur * permet de déstructurer la liste [2, 3, 4] en 3 nombres
# 2, 3 et 4
# liste[-1] = 5 (le dernier élément de la liste)
```

## Comment afficher des variables en plus du texte ?

Il est possible d'afficher le contenu de variables en plus de strings (chaines de caractères) avec la fonction print en Python. La méthode la plus couramment utilisée est probablement celles des chaînes de caractères littérales formatées (ou f-strings) mais il existe d'autres manières d'afficher des variables en plus du texte.

Voyons les 3 plus courantes ensemble.

## La concaténation avec print

En Python et avec la fonction print, on peut concaténer (coller des strings ensemble) avec l'opérateur + ou bien en passant plusieurs paramètres à print.

test004.py

```
un = 1
deux = 2
trois = 3

print('Un : ' + str(un) + ' deux : ' + str(deux) + ' trois : ' +
      str(trois))

print('Un', un, 'deux : ', deux, 'trois : ', trois)

##Tous les deux donnent Un 1 deux : 2 trois : 3 sur la sortie standard.
```

### Explications :

Quand on utilise l'opérateur +, il faut que tous les éléments soient du même type pour finalement être castés (transformés) en strings.

Dans l'exemple si dessus, j'utilise la fonction str() pour transformer nos nombres en objets de type string.

Quand on utilise les virgules, le séparateur par défaut est , , on a donc pas besoin de mettre des espaces comme quand utilise le +.

From:

<https://chanterie37.fr/fablab37110/> - Castel'Lab le Fablab MJC de Château-Renault

Permanent link:

[https://chanterie37.fr/fablab37110/doku.php?id=debuter\\_en\\_python:print&rev=1729434274](https://chanterie37.fr/fablab37110/doku.php?id=debuter_en_python:print&rev=1729434274)

Last update: 2024/10/20 16:24

