

Cheat sheet pour Python 3.

Type de variables

int	Entier	1 2 3 4 42
float	Réel	1.2 3.4 5. 42.0
str	Chaîne de caractères	"Texte" 'autre texte'
bool	Booléen	True False
tuple	Nuple	(1, 2) (1, 'a', 1.2)
list	List	[1, 2, 5] [1, 5, "e"]

Opérateur mathématique

+	Addition	1 + 2
+	Concatnation	"Hello" + "World"
-	Soustraction	3 - 4
*	Multiplication	5 * 6
*	Multiplication de str	"6" * 3
/	Division	7 / 8
//	Division entière	9 / 10

Opérateur de comparaison

==	Égal	1 == 2
!=	Différent	3 != 4
<	Plus petit	5 < 6
<=	Plus petit ou égal	7 <= 8
>	Plus grand	9 > 10
>=	Plus grand ou égal	11 >= 12
in	Est dedans égal	7 in ([1, 2, 3])

Fonctions Python

print	Affiche un texte dans la console	print("Hello!")
input	Renvoie la valeur entrée par utilisateur l'utilisateur sous forme de str	name = input("Name: ")
len	Renvoie la longueur d'une list, d'un tuple ou d'une str passé en paramètre.	len([1, 2, 3])

Méthode de list

append	Ajoute une valeur en fin de list.	ma_list.append(3)
insert	Insert une valeur l'index passé en paramètre. <i>insert(index, valeur)</i>	ma_list.insert(0, 'test')
remove	Retire la première occurrence de la valeur passée en paramètre.	ma_list.remove('t')
pop	Retire la valeur l'index passé en paramètre. Et renvoie cette valeur.	first = ma_list.pop(0)
sort	Trie la list par ordre croissant.	ma_list.sort()
reverse	Inverse le contenu de la list.	ma_list.reverse()

Random

randint	Renvoie un nombre aléatoire compris entre les deux int passés en paramètre.	bingo = randint(1, 50)
---------	---	------------------------

Import

Pour importer un module:

```
import <nom du module>
```

Pour importer une partie du module:

```
from <nom du module> import <partie du module>
```

Exemple:

```
import random  
from math import sin
```

Boucle arithmétique

```
for <variable> in <list, str ou range(>:
    <code contrôlé par la boucle>
```

Exemple:

```
number = 2
ma_list = []
```

```
for n in range(3):
    number = number + number
    lma_list.append(a)
```

```
for value in ma_list:
    print(value)
```

Condition

```
if <condition>:
    <code contrôlé par la condition>
elif <autre condition>:
    <code contrôlé par l'autre condition>
else:
    <code controlé exécuté si les autres coditions n'ont pas été remplie>
```

Il peut y avoir autant de elif que dsir, par contre il ne peut y avoir qu'un else qui est toujours mis la fin du bloc de condition.

Exemple:

```
import random

response = randint(1, 6)
test = int(input("tentative: "))
if response == tentative:
    print("Well done!")
elif response == 42:
    print("Cà c'est LA REPONSE")
else:
    print("Dommage")
```

Boucle logique

```
while <condition>:
    <code contrôlé par la boucle>
```

exemple:

```
result = []
```

```
while len(result) < 10:
    response = input("Donne une valeur:" )
    if response not in result:
        result.append(response)
```