

# Syntaxe Python - Memento

## Les modules

Import		Appel
<code>import math as mt</code>	→	<code>mt.sin(..)</code>
<code>from math import sin</code>	→	<code>sin(..)</code>

## Opérations basiques

- Commentaires :  
`# Commentaires`  
`"""Commentaires"""`
- Entiers : `+`, `-`, `*`, `//`, `%`, `**`
- Flottants : `+`, `-`, `*`, `/`, `**`
- Booléens : `not`, `or`, `and`
- Affectation :  
`variable = objet`
- Affichage : `print(variable)`
- Type : `type(variable)`

## Graphique

- Importer `matplotlib.pyplot`:
- Structure:  

```
f, ax = plt.subplots()
f.suptitle("Titre")
ax.set_xlabel("Legende X")
ax.set_ylabel("Legende Y")

ax.plot(x,y,label="Legende")
ax.errorbar(x,y,xerr=vX,
            yerr=vY)
ax.hist(datas,bins='rice')
# Options utiles:
# linestyle, linewidth
# marker, color

ax.grid()
ax.legend()
plt.show()
```

## Les listes

- `L = [3, "abc", 21., 42]`
- Sélection d'un élément : `L[0]` renvoie 3
- Sélection d'une partie `L[0:2]`, `L[2:]` renvoie `[3, "abc"]` et `[21., 42]`  
Indices inversés : `L[-1]` renvoie 42.
- Concaténation : `L1 + L2`
- Longueur: `len(L)` renvoie 4
- Ajout d'un élément: `L.append(6)`.  
`L` est modifiée: `[3, "abc", 21., 42, 6]`
- Suppression d'un élément: `L.pop(1)`.  
`L` est modifiée: `[3, 21., 42, 6]`

ATTENTION, les indices commencent à 0.

## Les chaînes de caractères

- `chaine = "Ceci est une chaine"`
- Sélection d'une partie : `chaine[0]`, `chaine[0:3]` renvoie "C" et "Cec"  
Même principe que pour les listes.
- Concaténation : `"Chaine 1" + "Chaine 2"` renvoie "Chaine 1Chaine 2"
- Longueur: `len("Chaine L")` renvoie 8
- Transformer en chaîne de caractère: `str(324)` renvoie "324"

## Les vecteurs numpy

- Importer numpy! `import numpy as np`
- Création:  

```
np.array(L) # L est une liste de flottants/entiers
np.arange(start, stop, step) # stop est exclus
np.linspace(start, stop, N) # stop est inclus
np.zeros(6) # np.ones(6)
```
- Sélection, Longueur : comme pour une liste
- Opération terme à terme : `L1 - L2`; `L1*3`, `np.sin(L)`,...  
ATTENTION `L1+L2` ne concatène pas  
Les vecteurs sont de taille fixe.

## Les fonctions

- Appel :  
`nom_fonction(arg1, arg2, ..)`
- Définition :  

```
def nom_fonction(arg1, ...):
    corps_fonction
    corps_fonction
    return var1, var2
```

## Structures conditionnelles

- Structure :  

```
if condition:
    corps_condition
elif condition: # Optionnel
    corps_condition
else: # Optionnel
    corps_condition
```
- Types conditions:  
Booléen, `==`, `>`, `<`, `>=`, `<=`, `in`, `not`

## Boucle for

- Structure :  

```
for var in obj_iterable:
    corps_boucle
    corps_boucle
    exterieur_boucle
```
- Objet itérable:  
Liste, Vecteur,  
`range(start, stop, pas)`

## Boucle while

- Structure :  

```
while condition:
    corps_boucle
    condition_modifiee
    corps_boucle
    exterieur_boucle
```

## Fonctions numpy usuelles

```
# Regression lineaire
# y = V[0]x + V[1]
V = np.polyfit(x, y, deg)
```

```
# Importation de donnees
c1 = np.loadtxt('chemin',
                skiprows=1, delimiter=',')
```

```
# Generation aleatoire
rd.uniform(a,b,N)
rd.normal(m,sigma,N)
```