Notions essentielles (niveau 2)

1) Les différents types de données.

- complexes
- tuples
- sets
- dictionnaires

```
1 + 1j      complex(1, 1)      2.5 + 1.5j
```

1) Les différents types de données.

- complexes
- tuples
- sets
- dictionnaires

```
(6, 2, 3)    ('Pierre', 'Paul')    ([1, 2], 45)
```

1) Les différents types de données.

- complexes
- tuples
- sets
- dictionnaires

`set()` *(vide)* `{1, 3, 8}` `{2 , 'Paul'}`

1) Les différents types de données.

- complexes
- tuples
- sets
- dictionnaires

{} {'Pierre' : 13, 'Paul' : 9} {(1,2) : 'Paul'}

1) Les différents types de données.

- complexes
- tuples
- sets
- dictionnaires

2) Les variables.

- Mutables
- Immuables
- Aliasing
- Copie

Listes

dictionnaires sets tableaux numpy

2) Les variables.

- Mutables
- Immuables
- Aliasing
- Copie

Entiers flottants booléens

chaines de caractères tuples

2) Les variables.

- Mutables
- Immuables
- Aliasing
- Copie

```
B = A
```

```
Var1 = Var2
```

2) Les variables.

- Mutables
- Immuables
- Aliasing
- Copie

```
L.copy()      list(A)      A[:]
```

```
import copy          B = copy.deepcopy(A)
```

2) Les variables.

- Mutables
- Immuables
- Aliasing
- Copie

3) Entrées/sorties.

- fichiers texte
- images

```
with open("exemple.txt", "r") as f:
    contenu = f.read()

# modification de contenu

with open("exemple.txt", "w") as f:
    f.write(contenu)
```

3) Entrées/sorties.

- fichiers texte
- images

```
import numpy as np
from PIL import Image

img = Image.open("image.png")
np_img = np.array(img)

# modification de np_img

img = Image.fromarray(np_img)

img.save("image_modifie.png")
```

3) Entrées/sorties.

- `fichiers texte`
- `images`

4) Opérations.

- complexes
- tuples
- sets
- dictionnaires

$+$ $-$ $*$ $/$

4) Opérations.

- complexes
- tuples
- sets
- dictionnaires

+ *

4) Opérations.

- complexes
- tuples
- sets
- dictionnaires

| &

4) Opérations.

- complexes
- tuples
- sets
- dictionnaires

4) Opérations.

- complexes
- tuples
- sets
- dictionnaires

4) Opérations.

Slicing

```
[start : stop : step]
```

4) Opérations.

Slicing

Listes chaines de caratères tuples

4) Opérations.

Slicing

listes L = [-1, 0, 2, 4, 8, 12]

4) Opérations.

Slicing

chaines de caractères $\quad$ T = "Hello world"

4) Opérations.

Slicing

tuples $\quad t = (0, 2, 3, 5, 8)$

5) Fonctions sur les variables.

- Sur les complexes.
- Sur les tuples.
- Sur les sets.
- Sur les dictionnaires.

```
z.real    z.imag    abs(z)    z.conjugate()
```

5) Fonctions sur les variables.

- Sur les complexes.
- Sur les tuples.
- Sur les sets.
- Sur les dictionnaires.

`len(t)` `list(t)` `max(t)` `sum(t)`

5) Fonctions sur les variables.

- Sur les complexes.
- Sur les tuples.
- Sur les sets.
- Sur les dictionnaires.

`len(s)` `list(s)` `max(s)` `sum(s)`

5) Fonctions sur les variables.

- Sur les complexes.
- Sur les tuples.
- Sur les sets.
- Sur les dictionnaires.

`s.add()` `s.remove()` `s1.union(s2)`

5) Fonctions sur les variables.

- Sur les complexes.
- Sur les tuples.
- Sur les sets.
- Sur les dictionnaires.

`d.keys()` `d.values()` `d.items()`

5) Fonctions sur les variables.

- Sur les complexes.
- Sur les tuples.
- Sur les sets.
- Sur les dictionnaires.

`d.pop()` `del d[item]` `d.update`

5) Fonctions sur les variables.

- Sur les complexes.
- Sur les tuples.
- Sur les sets.
- Sur les dictionnaires.

6) Structures de contrôle.

- elif
- Break
- Listes en compréhension.

```
if <condition 1> :
   bloc d'instructions 1
elif <condition 2> :
   bloc d'instructions 2
else :
   bloc d'instructions 3
```

6) Structures de contrôle.

- elif
- Break
- Listes en compréhension.

```
while <condition 1> :
   bloc d'instructions répété
   if <condition 2>
     break
```

6) Structures de contrôle.

- elif
- Break
- Listes en compréhension.

```
for k in <iterable> :
   bloc d'instructions répété
   if <condition>
     break
```

6) Structures de contrôle.

- elif
- Break
- Listes en compréhension.

```
L = [expression for variable in iterable]
```

```
L = [expression for variable in iterable if condition]
```

6) Structures de contrôle.

- elif
- Break
- Listes en compréhension.

7) Définition de fonctions.

- Argument par défaut.
- Effet de bords volontaires.
- Effets de bords involontaires

Exemple :

```
def subdivision(a, b, n = 50 ):
    L = [ a + k*(b-a)/n for k in range(n+1) ]
    return L
```

7) Définition de fonctions.

- Argument par défaut.
- Effet de bords volontaires.
- Effets de bords involontaires

Exemple :

```
def premier_dernier(L):
    a = L.pop(0)
    L.append(a)
```

7) Définition de fonctions.

- Argument par défaut.
- Effet de bords volontaires.
- Effets de bords involontaires

Exemple :

```
def couper(L):
    L.pop(0)
    L.pop()
    return L
```

7) Définition de fonctions.

- Argument par défaut.
- Effet de bords volontaires.
- Effets de bords involontaires

8) Modules.

- numpy.linalg
- scipy.stats
- numpy.random vs random

```
import numpy.linalg as la

        la.inv(A)      la.matrix_rank(A)

         la.eig(A)      la.eigvals(A)
```

8) Modules.

- numpy.linalg
- scipy.stats
- numpy.random vs random

```
from scipy.stats import norm, binom, poisson, t

    f = norm.pdf
    F = norm.cdf
    u = norm.ppf
    x = norm rvs
```

8) Modules.

- numpy.linalg
- scipy.stats
- numpy.random vs random

```
numpy.random.rand     vs  random.random
numpy.random.randint  vs  random.randint
numpy.random.normal   vs  random.gauss
numpy.random.uniform  vs  random.uniform
```

8) Modules.

- numpy.linalg
- scipy.stats
- numpy.random vs random