

IALAB
WORKSHOPS

Workshop IALab #1

Bases du python et de la manipulation de données



Sommaire

- Créer son environnement de travail
 - Suite Anaconda
 - Alternatives (Google Collab, éventuellement AWS)
- Le python 'basique'
 - Formater les données
 - Les boucles
 - Les fonctions
- Pandas
 - Importer des données
 - Manipuler les DataFrame
- Afficher des graphiques
 - Les différents types de graphiques
 - Embellir ses graphiques



Créer son environnement de travail

Garage ISEP
IA LAB - Workshop 0

La suite Anaconda





Les alternatives



Google Colab: Un jupyter Notebook en ligne (gratuit)

<https://colab.research.google.com>



AWS: Plateforme qui permet une grosse puissance de calcul (payant)



Les opérateurs

$a + b$	$2 + 4 == 6$	Addition
$a - b$	$2 - 4 == -2$	Soustraction
$a * b$	$2 * 4 == 8$	Multiplication
a / b	$2 / 4 == 0.5$	Division
$a ** b$	$2 ** 4 == 16$	Puissance
$a \% b$	$11 \% 3 == 2$	Modulo
$a // b$	$11 // 3 == 3$	Reste



Des types pour tous nos besoin

int

bool

str

tuple

float

complex

list

dict

et bien d'autres encore



Le typage dynamique fort

```
a = 6 #int  
b = 1.5 #float  
c = complex(3, 2.5) #complex  
  
d = (a/b + c) ** b  
#17.64121805806892 + 9.97309252801441j
```

On ne s'occupe pas des types



Interaction utilisateur

<code>print(value, ..., sep = ' ', end = '\n')</code>	<code>input(prompt)</code>
<p>value : éléments à afficher (séparé par des virgules)</p> <p>sep : le caractère(s) qui sépare chaque éléments</p> <p>end : caractère(s) systématiquement ajouté à la fin</p>	<p>prompt : chaîne de caractère imprimé avant d'attendre une entrée</p> <p>Renvoie toujours une chaîne de caractères, utiliser le cast pour convertir les types :</p> <pre>age = (int)(input("Quelle age avez vous ?"))</pre>
<pre>s = input("Quelle est la meilleur asso à l'Isep ? ") print("C'est...", s, sep='\n')</pre> <p>Quelle est la meilleur asso à l'Isep ? Garage Isep</p> <p>C'est...</p> <p>Garage Isep</p>	



Conditions

==	égalité
<	inférieur
>	supérieur
<=	inférieur ou égal
>=	supérieur ou égal
!=	différent

```
if (a == 0):  
    #faire quelque chose  
elif not (a < b):  
    #faire autre chose  
else:  
    #faire autre chose
```

not	non
and	et
or	ou

```
x = -b/2/a if d == 0 else ..  
.. ((-b-d**1.5)/2/a, (-b+d**1.5)/2/a) if d > 0 else ...
```



Boucle for 1/2

Boucle for, permet de parcourir un objet itérable

list	tuple	range	str
for x in <list>:	for x in <tuple>:	for x in <range>:	for c in <str>:
dictionnaire			
clefs	for k in <dict>.keys():		
valeurs	for v in <dict>.values():		
les deux	for k, v in <dict>.items():		



Boucle for, exemple

```
garage = {  
    "ialab": ["Lucas", "Pierre-Louis", "Frank", "Jérémie", "Olivier", "Théophile"],  
    "cyberlab": ["Julien", "Salim", "Raphaël", "Martin"]  
}
```

```
for pole, membres in garage.items():  
    print("\nMembres du", pole, end=" : ")  
    for membre in membres:  
        print(membre, end=' ')
```

```
Membres du ialab : Lucas Pierre-Louis  
Frank Jérémie Olivier Théophile  
Membres du cyberlab : Julien Salim  
Raphaël Martin
```



Boucle for 2/2

`range(start, stop, step)`

Renvoie une séquence de nombre de start inclus à stop exclu avec un pas de step

`break`

Permet de stopper la boucle avant qu'elle n'ait parcouru tous les éléments

`continue`

Passe à la prochaine itération sans "terminer" celle en cours

```
for i in range(100):  
    if i % 3 == 0: continue  
    if i > 10: break  
    print(i, end=' ')
```

Output :

1 2 4 5 7 8 10



Boucle while

Boucle while, répéter un code tant que...

continue

Passe à la prochaine itération sans "terminer" celle en cours

break

Permet de quitter la boucle sans vérifier la condition

else

Permet d'exécuter du code quand la boucle termine naturellement (sans break)

```
while <condition>:  
    #faire des choses
```



Boucle while, exemple

```
n = randrange(10)
x = -1
essai = 5
while x != n:
    if essai == 0:
        print("Perdu !")
        break
    x = (int)(input("Deviner : "))
    if x > 9 or x < 0: continue
    essai -= 1
else:
    print("Gagné !")
```

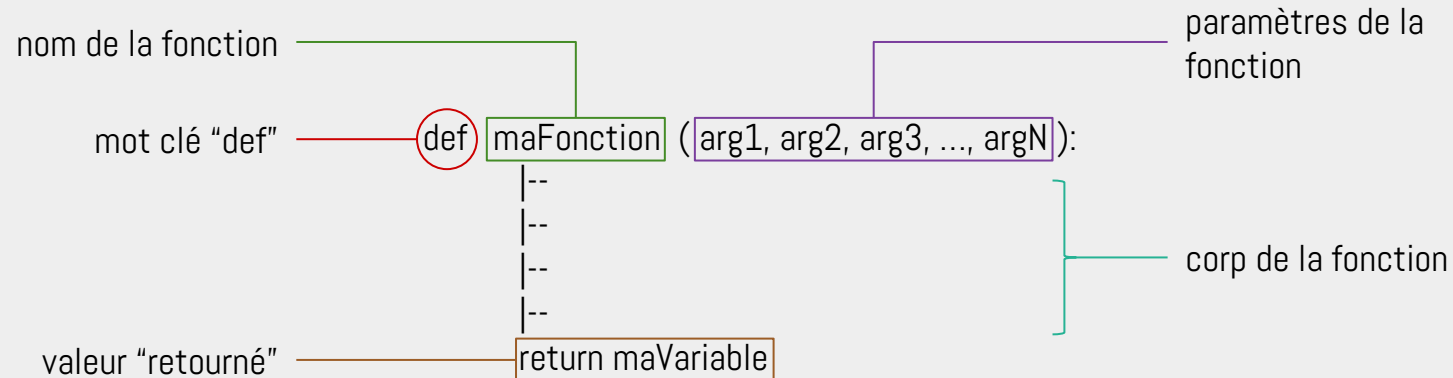
```
Deviner : 5
Deviner : 4
Deviner : 2
Deviner : 8
Gagné !
```

```
Deviner : 1
Deviner : 5
Deviner : 8
Deviner : 7
Deviner : 9
Perdu !
```



Fonctions

Evitons la redondance, les fonctions...





Fonctions exemple 1

```
def fact(n):  
    rtn = 1  
    for i in range(1, n+1):  
        rtn *= i  
    return rtn
```

```
print(fact(5))
```

Output : 120



Fonctions récursivité

```
def fact(n):  
    rtn = 1  
    for i in range(1, n+1):  
        rtn *= i  
    return rtn
```

```
def fact(n):  
    if n == 1:  
        return n  
    return n*fact(n-1)
```

```
def fact(n):  
    return n if n == 1 else n*fact(n-1)
```



Fonctions arguments

Si on ne connaît pas le nombre d'argument que doit recevoir la fonction ?

Le paramètre `*<paramètre>` permet de convertir tous les arguments suivant en une list

Le paramètre `**<paramètre>` permet de convertir tous les arguments suivant en un dictionnaire

```
def maFamille(**membres):  
    print("Ma famille :")  
    for relation, membre in membres.items():  
        print(relation, ":", membre)  
  
maFamille(père="Pierre", mère="Mathilde", frère="Zac",  
          soeur="Petronille")
```

```
Ma famille :  
père : Pierre  
mère : Mathilde  
frère : Zac  
soeur : Petronille
```



- Numpy arrays
- Pandas DataFrame
- Manipulation et extraction de données
- Lecture de fichiers



Numpy arrays

Le numpy array permet de représenter les données sous forme de tableau et utiliser les opérations vectorielle et matricielle

```
vector = np.array([x])  
  
matrix = np.array([x,y])
```



Pandas dataframe

Pandas est une librairie basée sous numpy

Permet d'utiliser les *DataFrame*

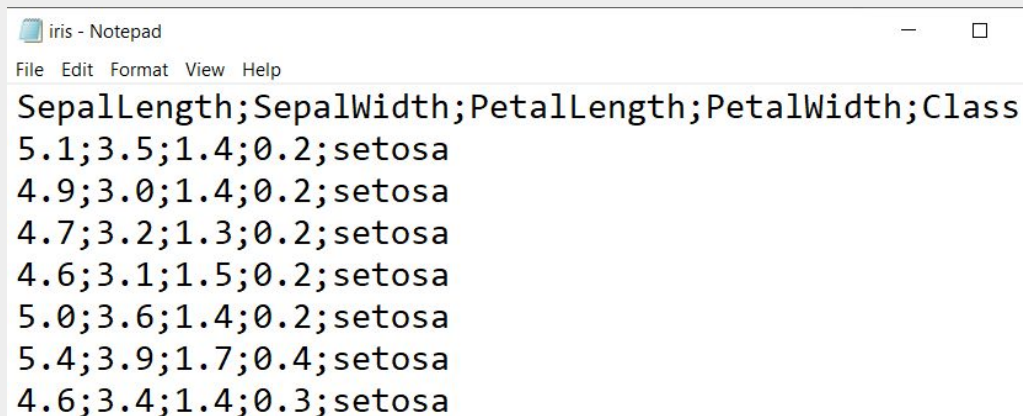
```
df = pd.DataFrame(data, columns=StringArray, index=StringArray)
```



Importer des données depuis un fichier - 1

Fichiers .csv : "Comma-separated
values"

Attention à bien prendre en
compte la composition du fichier



```
iris - Notepad
File Edit Format View Help
SepalLength;SepalWidth;PetalLength;PetalWidth;Class
5.1;3.5;1.4;0.2;setosa
4.9;3.0;1.4;0.2;setosa
4.7;3.2;1.3;0.2;setosa
4.6;3.1;1.5;0.2;setosa
5.0;3.6;1.4;0.2;setosa
5.4;3.9;1.7;0.4;setosa
4.6;3.4;1.4;0.3;setosa
```



Importer des données depuis un
fichier - 2

```
iris = pd.read_csv(fichier, sep=";")
```



Obtenir des informations sur les données

Méthodes à utiliser lors de la découverte d'un dataset

Fonctions pouvant également être appelée sur des colonnes uniquement

```
df.shape()
```

```
df.columns # pas une fonction
```

```
df.info()
```

```
df.describe()
```




Accéder aux données - 1

Possibilité d'isoler des attributs et d'obtenir des informations relatives aux colonnes

```
df["Colonne"] ou df.colonne
```

```
df.colonne.sort_values()
```

```
df.sort_values(by="colonne").head()
```

```
df.loc[df["Colonne"]=="cible"]
```



Accéder aux données - 2

Possibilité d'obtenir d'affiner la sélection

“Combien de personnes on plus de 25 ans”

“Combien on plus de 25 ans et habite à Paris”

etc.

```
(df.['Colonne']=="cible").value_counts()  
  
((df.['Colonne']=="cible") &&  
(df.['Colonne']>X)).value_counts()
```



Manipuler les données

Copier certaines colonnes d'un df

```
new_df = df[[colonnes]].copy()
```

Convertir les données d'une colonne

```
map_dict = {"label1":0,"label1":1,"label2":2}  
  
Y[colonne] = Y[colonne].map(map_dict)
```



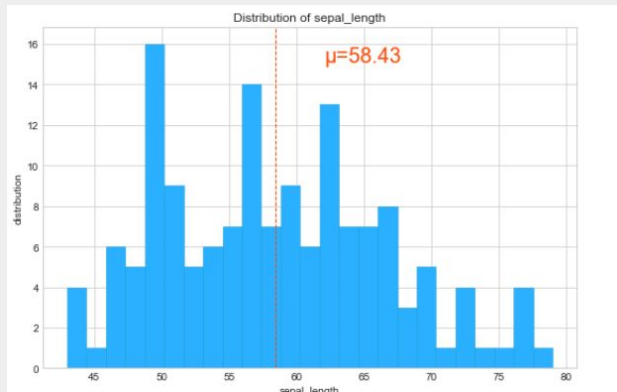
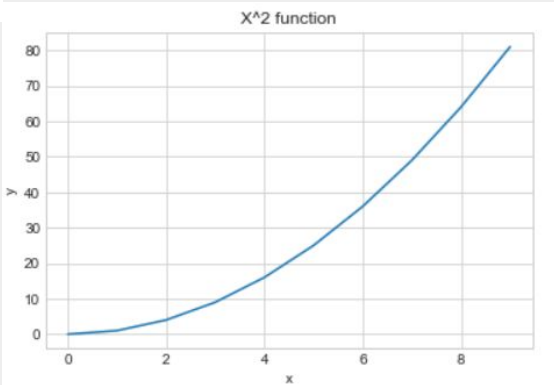
Les graphiques avec plt et seaborn

Garage ISEP
IA LAB - Workshop 0



heatmap

plot



hist



IALab - Workshops



<https://www.facebook.com/garageisep/>