



# Initiation à la programmation en Python pour écologues

Souleymane BAH  
bsouleymane@gmail.com

<https://souleymanebah.wordpress.com/>

Vendredi 14 avril 2023

Mis à jour en juin 2024

# PLAN

- Présentations formateur et participants
- Python et ses applications
- TP1 et TP2
- Opérateurs, Conditions, Boucles
- Procédure installation modules
- Google Colab
- TP3
- Iris
- TP4 (Pandas)
- TP5 (Matplotlib et Seaborn)
- Ressources pour aller plus loin

# Présentations

- **Souleymane BAH**
  - Ingénieur en téléinformatique
  - Doctorant en informatique : IA & agriculture de précision
- **Et vous ?**

# Le langage Python et ses applications

- 20 février 1991, Guido van Rossum
- Multiplateforme : **Windows**, Linux, macOS, Android, iOS
- Indentation par blocs
- Python 2 vs **Python 3**
  - Python 2.7 : éléments obsolètes et redondants, fin du support en 2020
  - Python 3 : décembre 2008, plus de compatibilité ascendante
- Utilisation : web, intelligence artificielle, cybersécurité, **calcul scientifique**
- Utilisateurs notables : Google, Facebook, NASA, ...

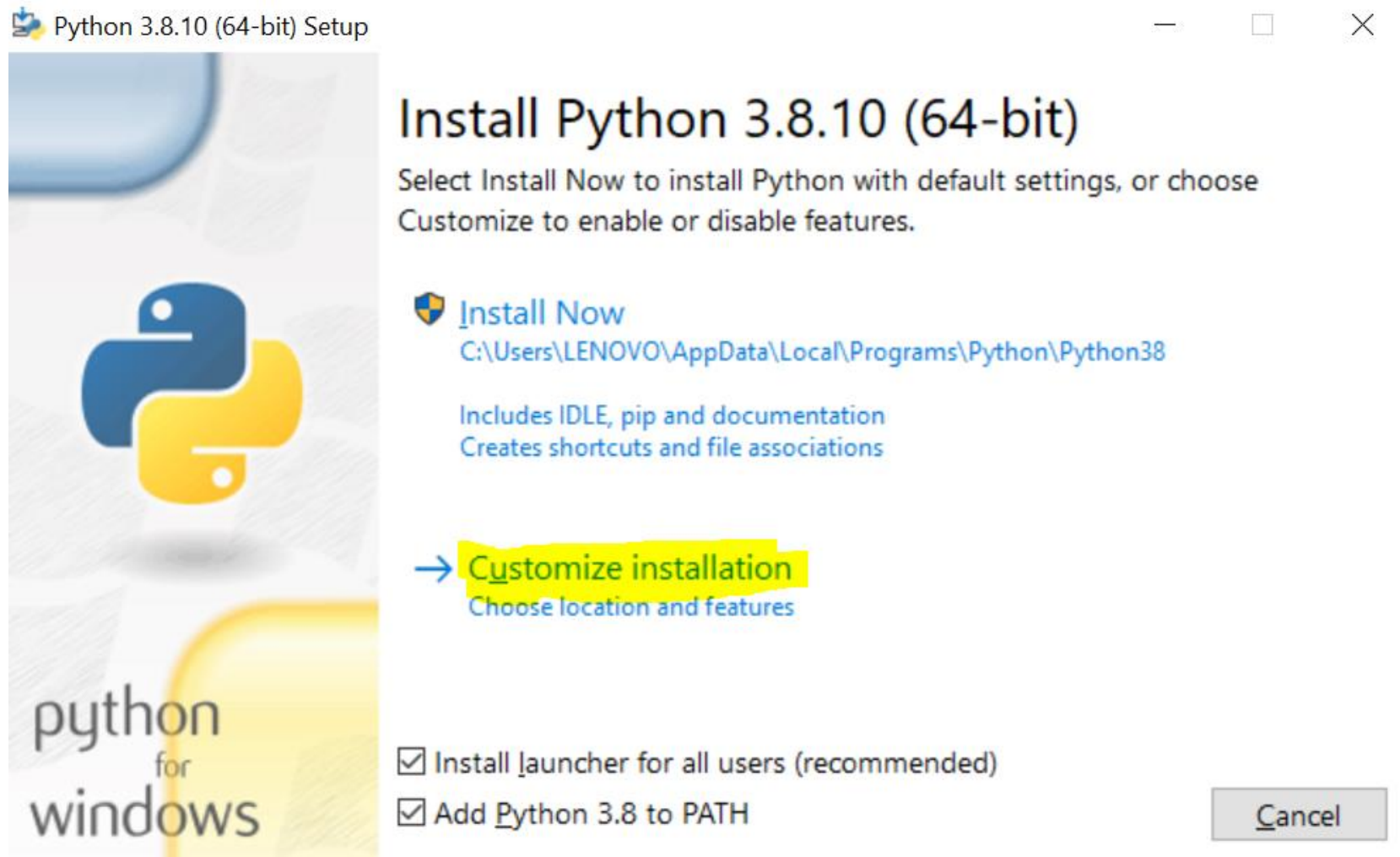
# TP1 : Installation des outils de travail 1/7

- Windows 7 : <= Python 3.8.10
  - 32 bits : <https://www.python.org/ftp/python/3.8.10/python-3.8.10.exe>
  - 64 bits : <https://www.python.org/ftp/python/3.8.10/python-3.8.10-amd64.exe>
- Windows 8 et Windows 10 : Python 3.11.0
  - 32 bits : <https://www.python.org/ftp/python/3.11.0/python-3.11.0.exe>
  - 64 bits : <https://www.python.org/ftp/python/3.11.0/python-3.11.0-amd64.exe>
- **Python 3.8.10**
  - 32 bits : **python-3.8.10**
  - 64 bits : **python-3.8.10-amd64**

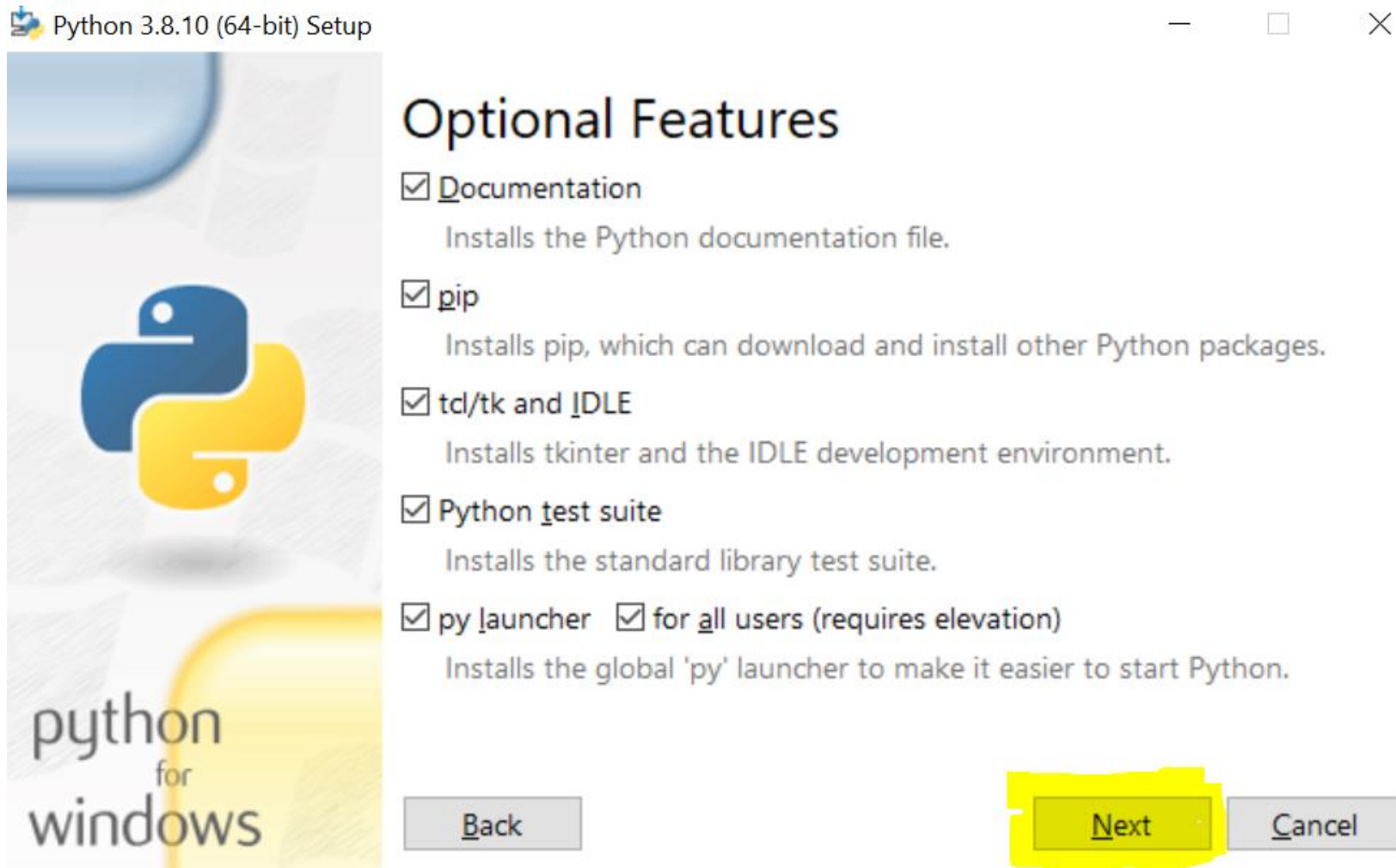
# TP1 : Installation des outils de travail 2/7



# TP1 : Installation des outils de travail 3/7

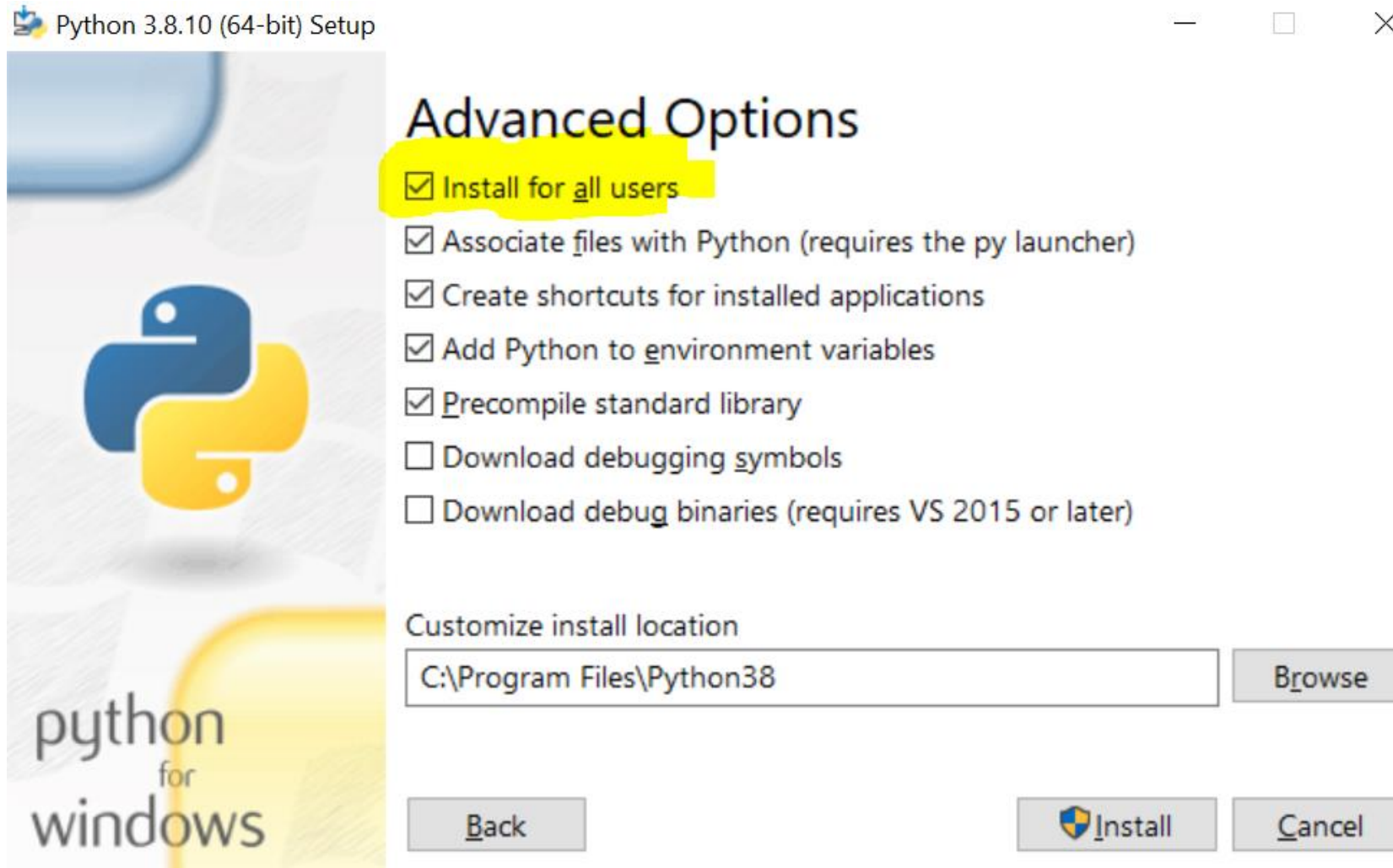


# TP1 : Installation des outils de travail 4/7

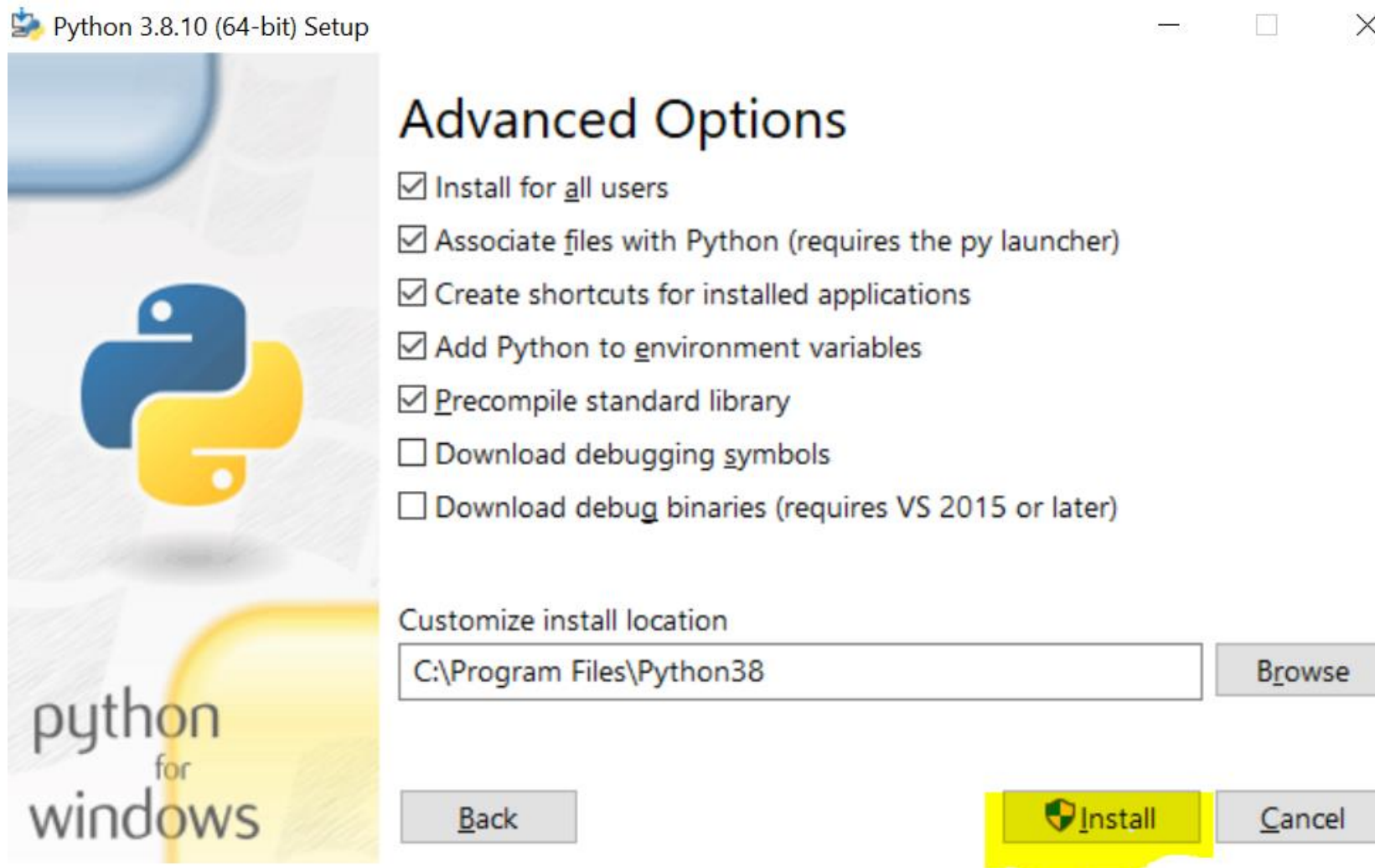




# TP1 : Installation des outils de travail 5/7



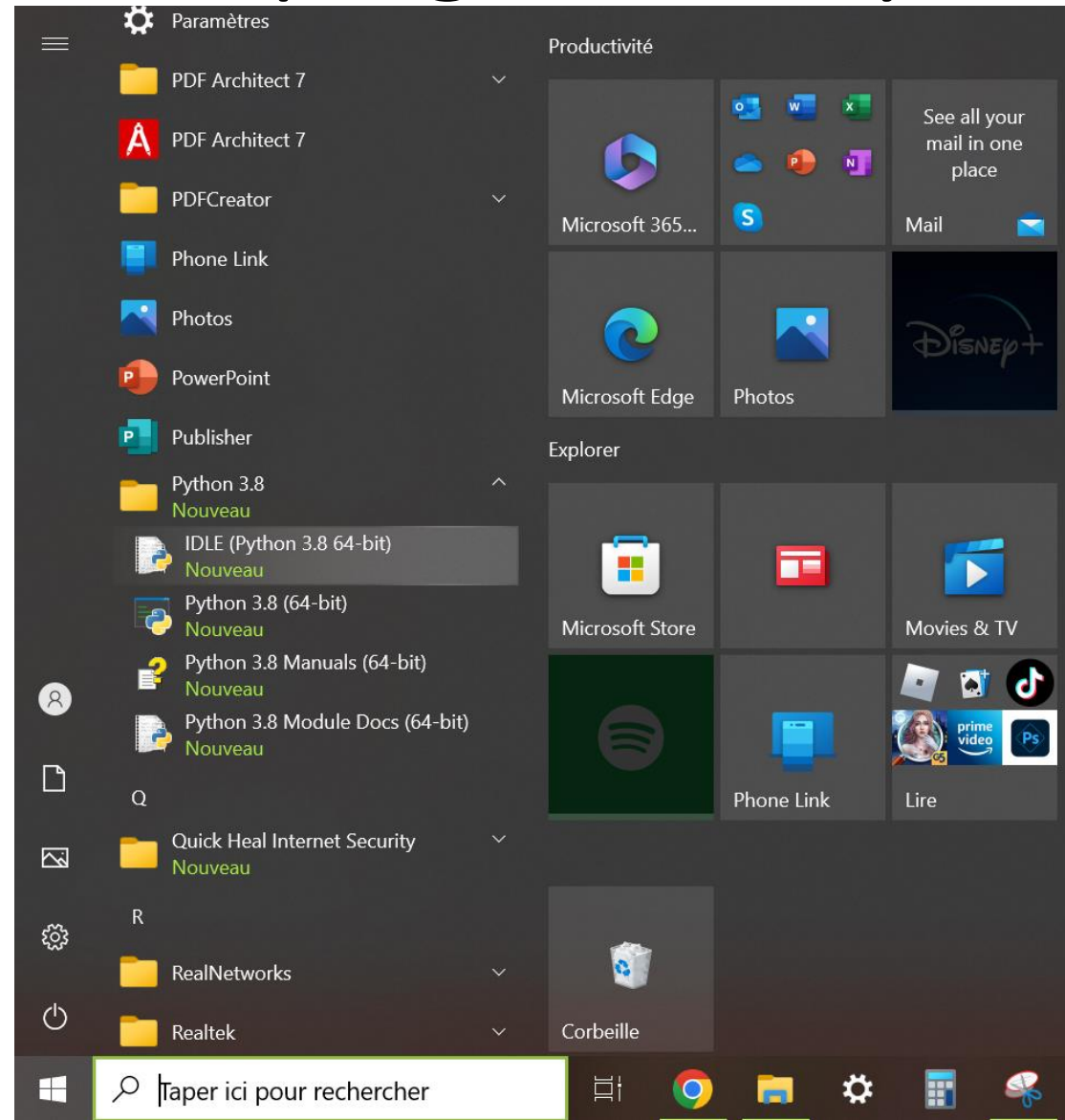
# TP1 : Installation des outils de travail 6/7



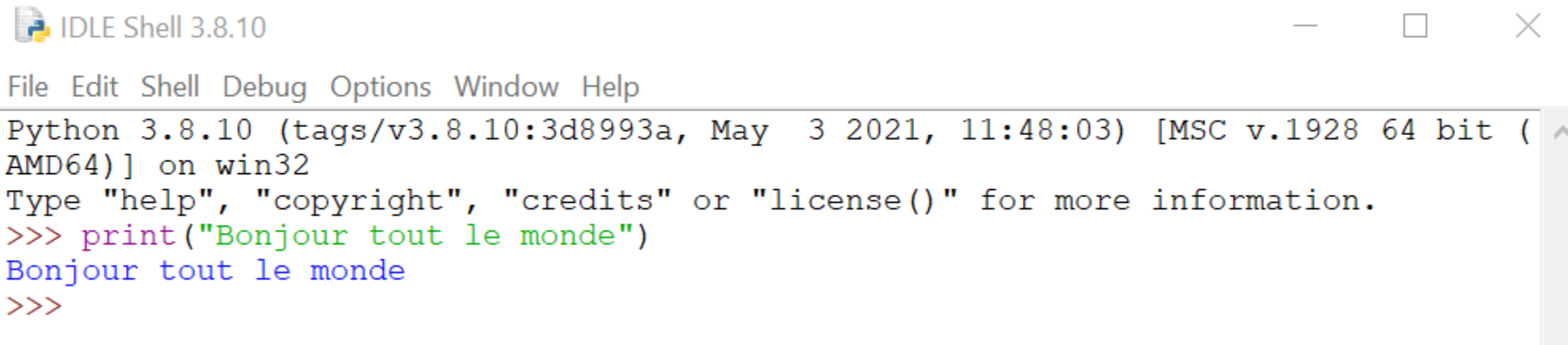
# TP1 : Installation des outils de travail 7/7



# TP2 : Mon premier programme Python 1/4



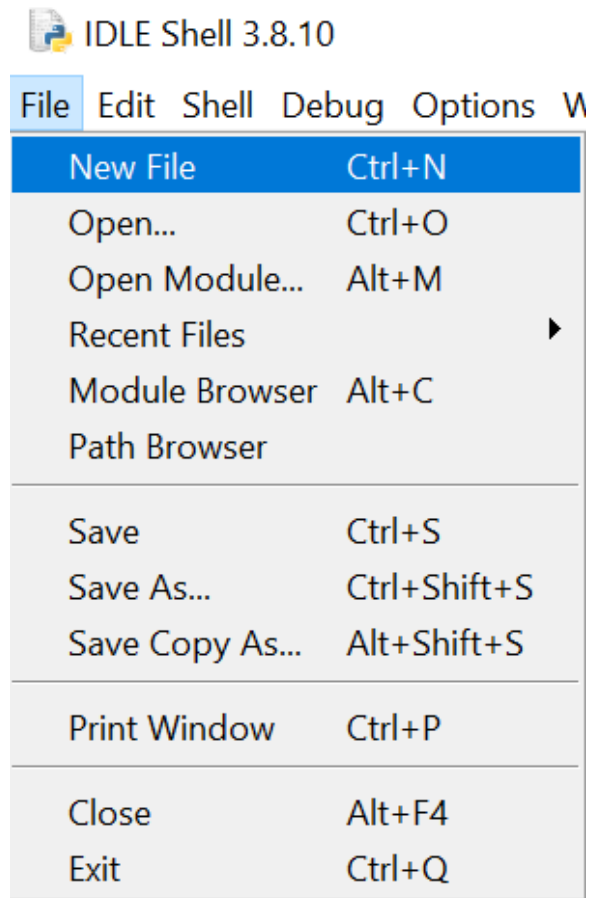
# TP2 : Mon premier programme Python 2/4




The screenshot shows the Python IDLE Shell 3.8.10 window. The title bar reads "IDLE Shell 3.8.10". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following content: "Python 3.8.10 (tags/v3.8.10:3d8993a, May 3 2021, 11:48:03) [MSC v.1928 64 bit (AMD64)] on win32", "Type 'help', 'copyright', 'credits' or 'license()' for more information.", and a successful execution of the command `>>> print("Bonjour tout le monde")` resulting in the output "Bonjour tout le monde". The prompt `>>>` is shown again on the next line. A vertical scrollbar is visible on the right side of the text area.

```
IDLE Shell 3.8.10
File Edit Shell Debug Options Window Help
Python 3.8.10 (tags/v3.8.10:3d8993a, May 3 2021, 11:48:03) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Bonjour tout le monde")
Bonjour tout le monde
>>>
```

# TP2 : Mon premier programme Python 3/4



# TP2 : Mon premier programme Python 4/4

 \*untitled\*

File Edit Format Run Options Window Help

```
print("Bonjour tout le monde")|
```

- Save : CTRL + S ; nommer le fichier
- Run module : F5

# Opérateurs 1/5

## Types de données et variables

- Mots réservés Python
  - Signification particulière
  - Ne peuvent être utilisés comme nom de variable ou de constante

and	as	assert	break	class	continue	def	del
elif	else	except	exec *	finally	for	from	global
if	import	in	is	lambda	not	or	pass
print *	raise	return	try	while	with	yield	
True **	False **	None **					

- \* Python 2 uniquement
- \*\* Python 3 uniquement



# Opérateurs 2/5

## Types de données et variables

### 4 Types primitifs

- Numériques
  - Entiers (integer)
  - Virgules (float)
- Chaîne de caractères (strings)
- Booléens (Boolean)

### Pratique

- `>>> type(2023)`
- `>>> type(0)`
- `>>> type(10.2)`
- `>>> type(0.0)`
- `>>> type('bonjour')`
- `>>> type("bonjour")`
- `>>> type(True)`
- `>>> type(False)`

# Opérateurs 3/5

## Types de données et variables

### Commentaires

- # En début ou en fin de ligne

### Pratique

- `>>> # mon commentaire`
- `>>> print("test") # commentaire de fin de ligne`

# Opérateurs 4/5

## Types de données et variables

### Variables et constantes

- Variable : association entre identifiant et valeur
- Déclaration :
  - `>>> nom_variable = valeur`
- Nom variable
  - Donner nom significatif :
    - `>>> x = "Bonjour"`
    - `>>> message = "Bonjour"`
  - Ne peut pas utiliser un mot réservé
  - Ne peut pas commencer par un chiffre
  - Ne peut pas contenir un espace
  - Sensible à la casse
  - Pas d'accents, ni de tirets
  - Suite de minuscules séparés par `_` :
- Constante : valeur "figée"
  - Idem + Suite de MAJUSCULES séparés par `_`

### Pratique

- `>>> message_accueil = "Bienvenue"`
- `>>> print(message_accueil)`
- `>>> poids = 15`
- `>>> Poids = 20`
- `>>> print(poids)`
- `>>> print(Poids)`
- `>>> VALEUR_SEUIL = 10.2`
- `>>> print(VALEUR_SEUIL)`

# Opérateurs 5/5

## Liste des opérateurs

- Somme :  $x + y$
- Différence :  $x - y$
- Multiplication :  $x * y$
- Division :  $x / y$
- Partie entière :  $x // y$
- Reste :  $x \% y$
- Puissance :  $x ** y$

## Pratique

1.  $x = 13$
2.  $y = 5$
3.  $somme = x + y$
4. `print(somme)`
5. Faire de même pour les autres opérateurs

# Conditions 1/2

**if** *condition1*:

instruction\_1  
instruction\_2  
...

**elif** *condition2*:

instruction\_3  
instruction\_4  
...

**elif** *condition3*:

instruction\_5  
instruction\_6  
...

**else**:

instruction\_n  
instruction\_n+1  
...

- **Associer plusieurs conditions avec opérateurs logiques :**
  - ET : and
  - OU : or
  - NON : not

## EXEMPLE

- x = True
- y = False
- test\_1 = x and y
- print(test\_1)
- test\_2 = x or y
- print(test\_2)
- test\_3 = not x
- print(test\_3)
- z = True
- test4 = x and z
- print(test4)

# Conditions 2/2

## Comparaison nombres

- égalité :  $x == y$  (différent de  $=$ )
- non égalité :  $x != y$
- inférieur :  $x < y$
- supérieur :  $x > y$
- inférieur ou égal :  $x <= y$
- supérieur ou égal :  $x >= y$

## Pratique

- $a = 5$
- $b = 5$
- `print(a == b)`
- `print(a != b)`
- Mettez une valeur dans la variable  $x$
- Mettez une autre valeur dans la variable  $y$
- Comparez  $x$  et  $y$
- Affichez le message :
  - **$x$  est plus grand que  $y$**
  - **sinon  $y$  est plus grand que  $x$**

**PAUSE 15 mn**

# Boucles 1/2

## for

- Répéter un nombre de fois **connu par avance**

for *élément* in *séquence*:

    instruction\_1

    instruction\_2

    ...

    instruction\_n

- séquence : liste, tuple, dictionnaire, chaîne de caractères ou **range**
  - range(n) # séquence de 0 à n - 1
  - range(m, n) # séquence de m à n - 1
  - range(m, n, i) # par pas de i (+/-)

## Pratique : for.py

1. Afficher une séquence de 0 à 5
2. Afficher une séquence de 1 à 15
3. Afficher une séquence de 1 à 15 par pas de 2
4. Répétez l’affichage du message **Bonjour à tous** 5 fois



# Boucles 2/2

## while

- Répéter un nombre de fois **tant que condition remplie**

*while condition:*

instruction\_1

instruction\_2

...

instruction\_n

- **Faire attention aux boucles infinies : CTRL + C → STOP**

## Pratique while.py

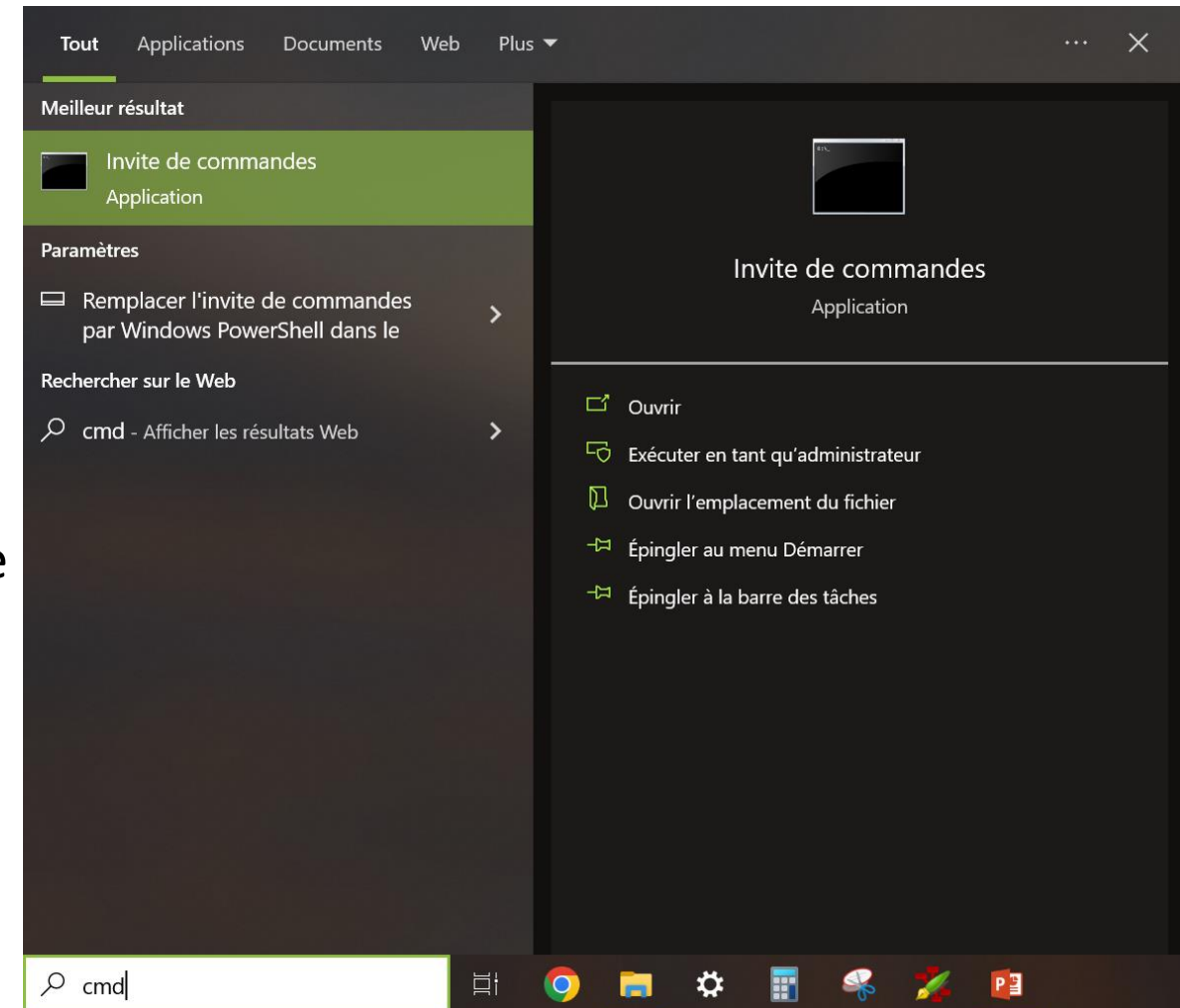
1. Initialisez la variable i avec la valeur 75
2. Faites une boucle qui affiche la valeur de i tant que  $i \leq 100$
3. **Dans la boucle augmentez la valeur de i d'une unité ( $i = i + 1$ )**
4. Testez

# Installation de modules

- Ajout de fonctionnalités
  - Menu démarrer puis CMD
  - **Installation :**
    - >>> `pip install nom_module`
- **Utilisation (code Python)**
  - `import nom_module`
  - Utilisation suivant documentation module
- **Connexion internet nécessaire**

## EXEMPLE

- Installez le module geoformat



QUESTIONS ?

**PAUSE : REPRISE À 14h00**

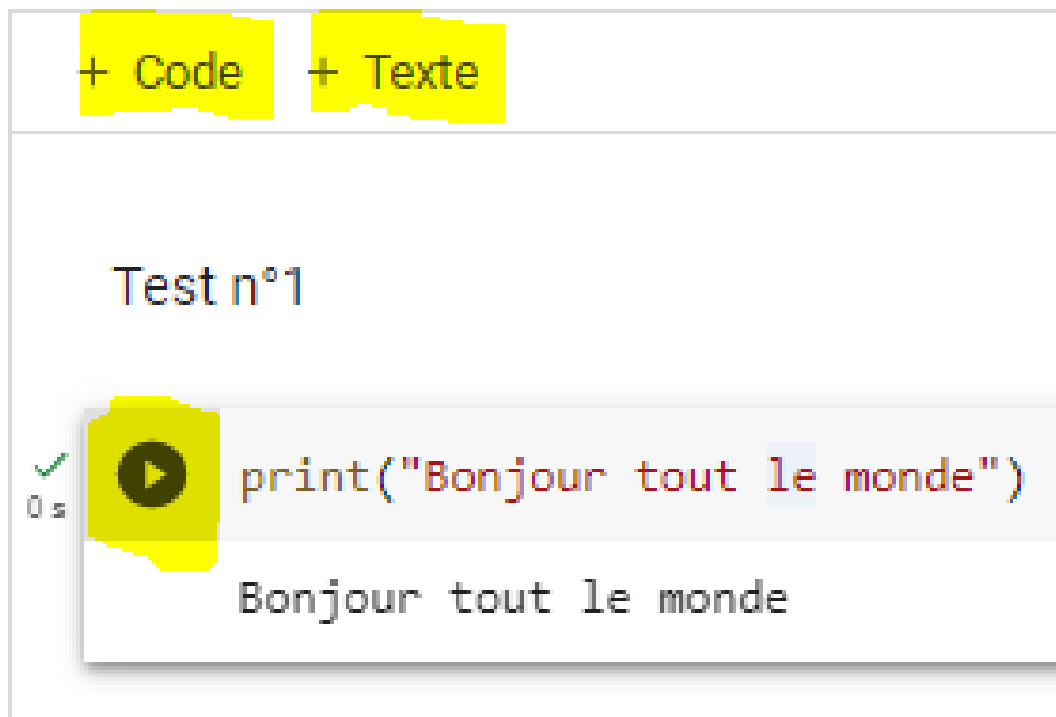
# Présentation de Google Colab

- Ecrire et exécuter code Python dans un simple navigateur web
- Puissance de calcul en ligne et gratuite
- Bibliothèques préinstallées
- Facile à partager
- Accessible depuis n'importe où
- Orienté Machine Learning
- Version payante pour plus de puissance



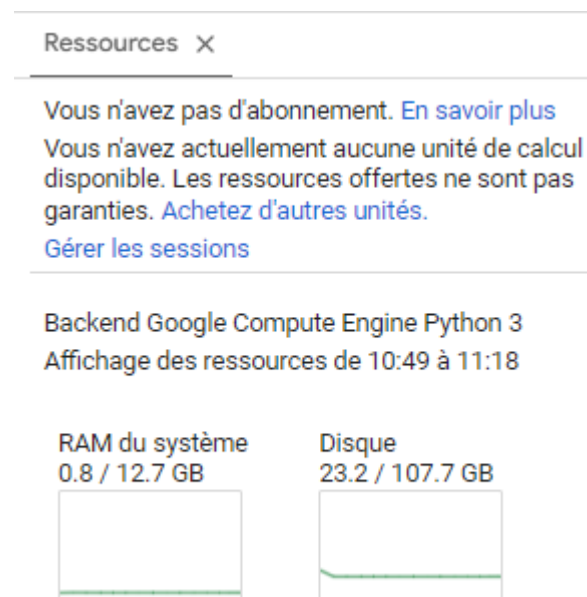
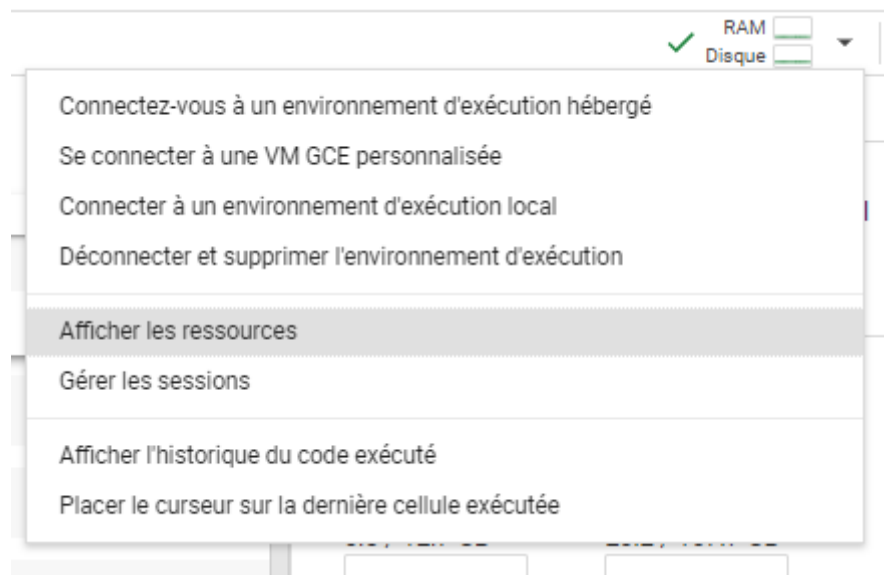
# TP3 : Prise en main de Colab 1/6

- <https://colab.research.google.com/> → Nécessite un compte Google
- Ecrire et exécuter code :
  - Fichier → Nouveau notebook
  - Fichier → Renommer
  - Code → CTRL + M + B
  - Exécuter → CTRL + Entrée



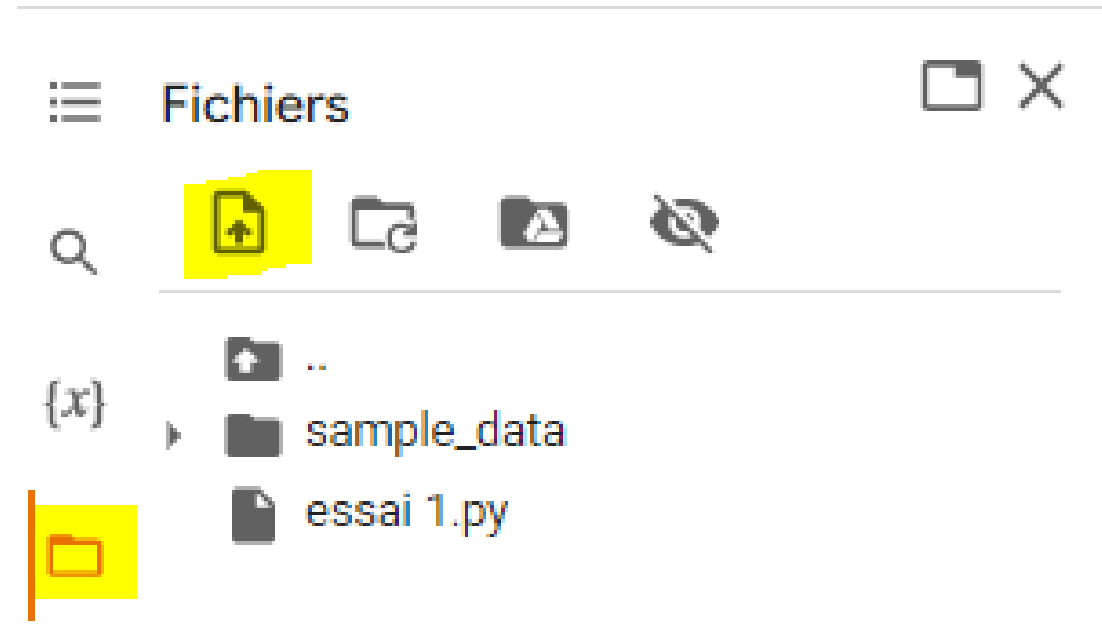
# TP3 : Prise en main de Colab 2/6

- Afficher les ressources



# TP3 : Prise en main de Colab 3/6

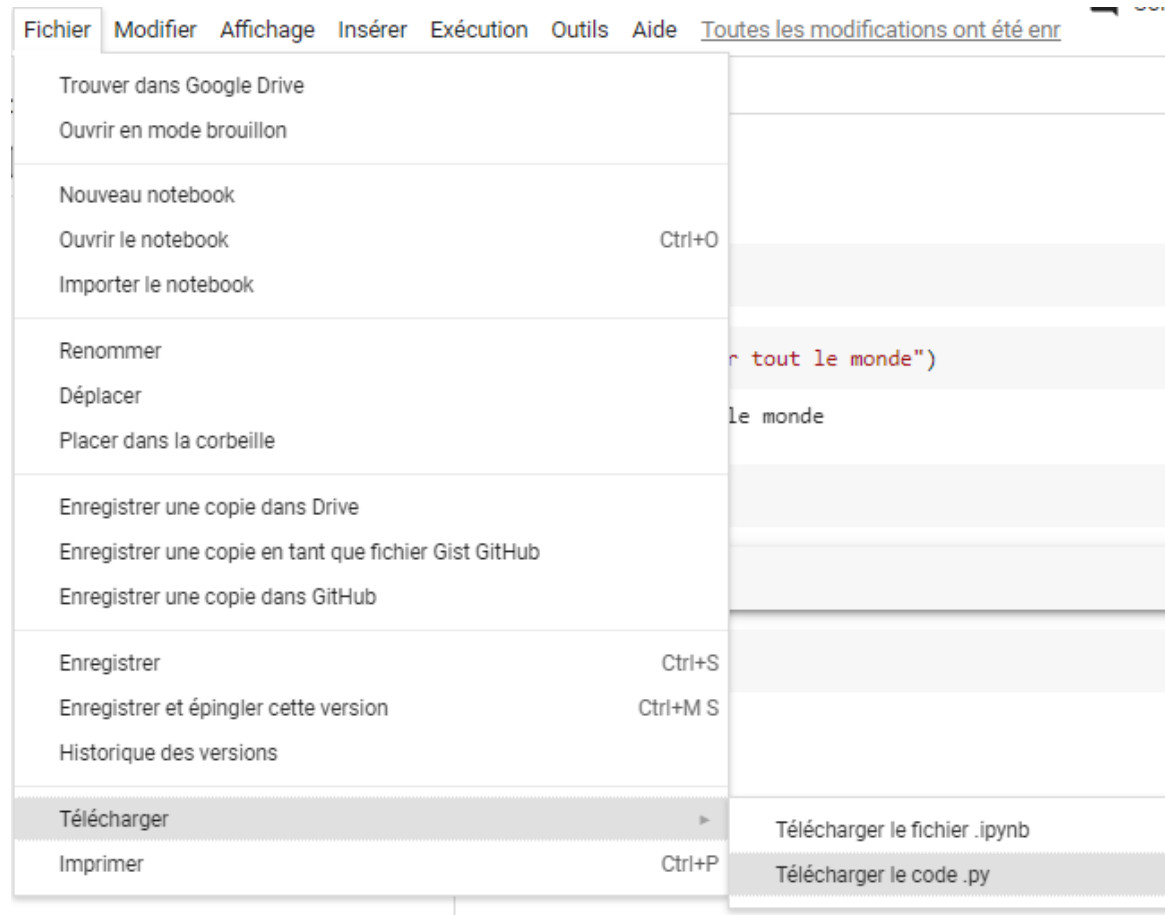
- Importer fichiers





# TP3 : Prise en main de Colab 4/6

- Télécharger fichiers



# TP3 : Prise en main de Colab 5/6

- Installer module (non préinstallé)
  - !pip install geoformat

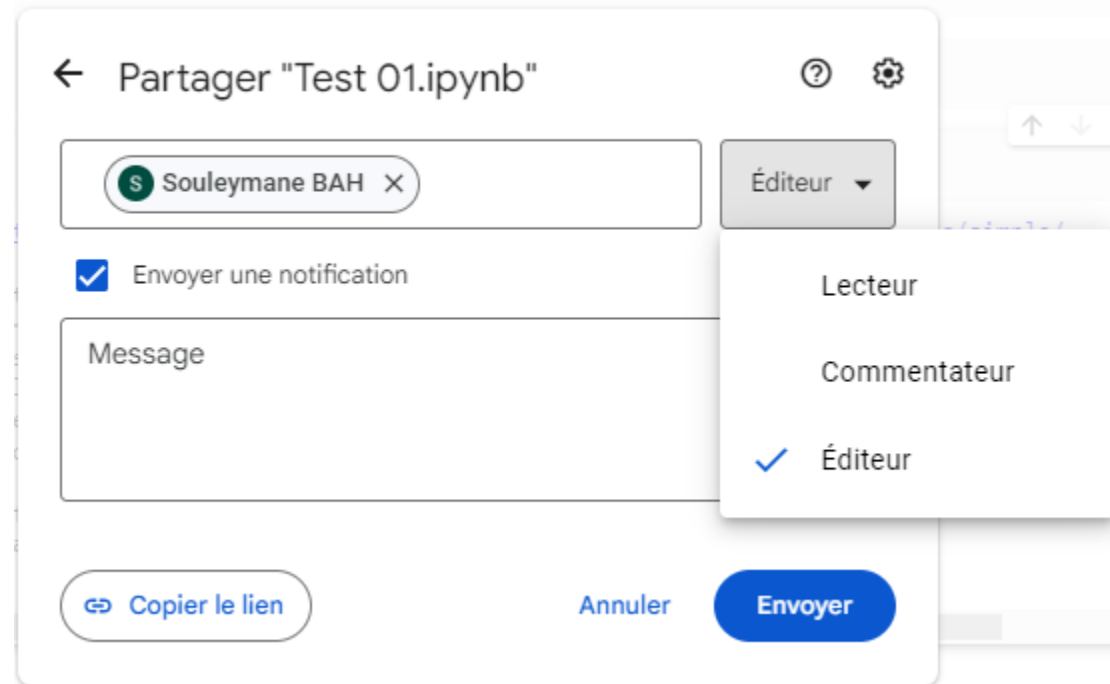
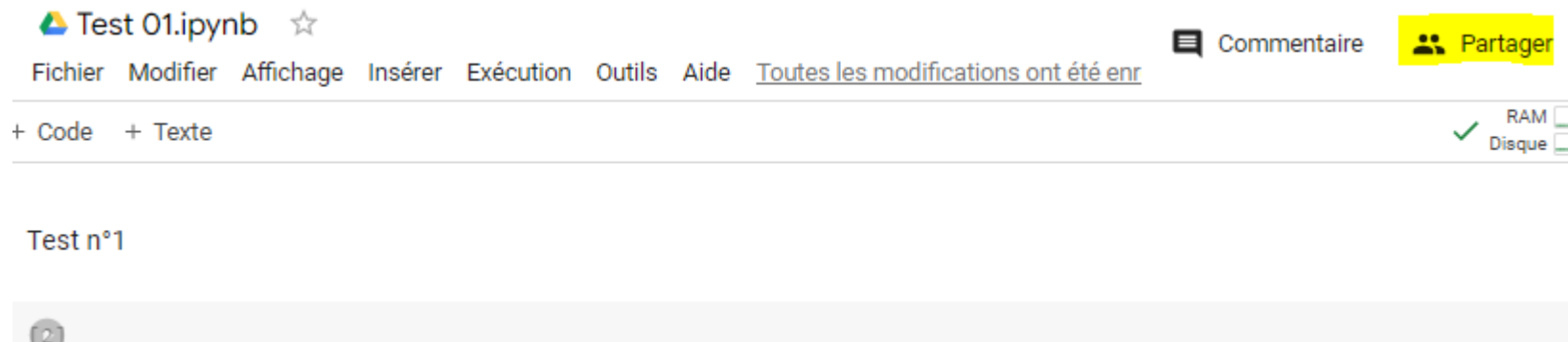


```
✓ 10 s ▶ !pip install geoformat

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting geoformat
  Downloading geoformat-20230116.tar.gz (3.1 MB)
    _____ 3.1/3.1 MB 4.9 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: geoformat
  Building wheel for geoformat (setup.py) ... done
  Created wheel for geoformat: filename=geoformat-20230116-py3-none-any.whl size=138489 sha256=0123d6a38d47de4c34cdbc95c617
  Stored in directory: /root/.cache/pip/wheels/87/ea/7d/070f9654349f26379de5ff7183213a840163bc60e067218ff2
Successfully built geoformat
Installing collected packages: geoformat
Successfully installed geoformat-20230116
```

# TP3 : Prise en main de Colab 6/6

- Partager



# Présentation du dataset Iris

- 1936, Ronald Fisher & Edgar Anderson
- Variations de morphologie (L&I) des fleurs d'iris de trois espèces,  $50 \times 3 = 150$
- Ouvrez le fichier **iris-Excel** pour avoir un aperçu

**iris setosa**



petal

sepal

**iris versicolor**



petal

sepal

**iris virginica**



petal

sepal

# TP4 : Analyse du jeu de données Iris avec Pandas 1/4

- Créez un notebook et Importez le fichier **iris.csv**
- Saisissez puis exécutez le code suivant

```
import pandas as pd
```

```
# Lecture du fichier
```

```
df = pd.read_csv("iris.csv")
```

```
# 5 premières et dernières lignes  
df
```

df					
	sepal.length	sepal.width	petal.length	petal.width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

# TP4 : Analyse du jeu de données Iris avec Pandas 2/4

- Informations sur le jeu de données 1/2
- Entrez puis exécutez le code suivant (nouvelle cellule pour chaque ligne)

# Colonnes et types  
df.info()

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype  
---  -
0   sepallength      150 non-null   float64
1   sepalwidth       150 non-null   float64
2   petallength      150 non-null   float64
3   petalwidth       150 non-null   float64
4   class            150 non-null   object  
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

# Résumé statistique  
df.describe()

```
df.describe()

      sepallength  sepalwidth  petallength  petalwidth
count    150.000000    150.000000    150.000000    150.000000
mean         5.843333         3.054000         3.758667         1.198667
std          0.828066         0.433594         1.764420         0.763161
min          4.300000         2.000000         1.000000         0.100000
25%          5.100000         2.800000         1.600000         0.300000
50%          5.800000         3.000000         4.350000         1.300000
75%          6.400000         3.300000         5.100000         1.800000
max          7.900000         4.400000         6.900000         2.500000
```

# TP4 : Analyse du jeu de données Iris avec Pandas 3/4

- Informations sur le jeu de données 2/2
- Entrez puis exécutez le code suivant (nouvelle cellule)

```
# Valeurs manquantes  
df.isnull().sum()
```

```
sepallength    0  
sepalwidth     0  
petallength    0  
petalwidth     0  
class          0  
dtype: int64
```

# TP4 : Analyse du jeu de données Iris avec Pandas 4/4

- Matrice de corrélation
- Entrez puis exécutez le code suivant (nouvelle cellule)

# On ne considère que les colonnes numériques

```
df2 = df[["sepalength", "sepalwidth", "petallength", "petalwidth"]]  
df2.corr(method='pearson')
```

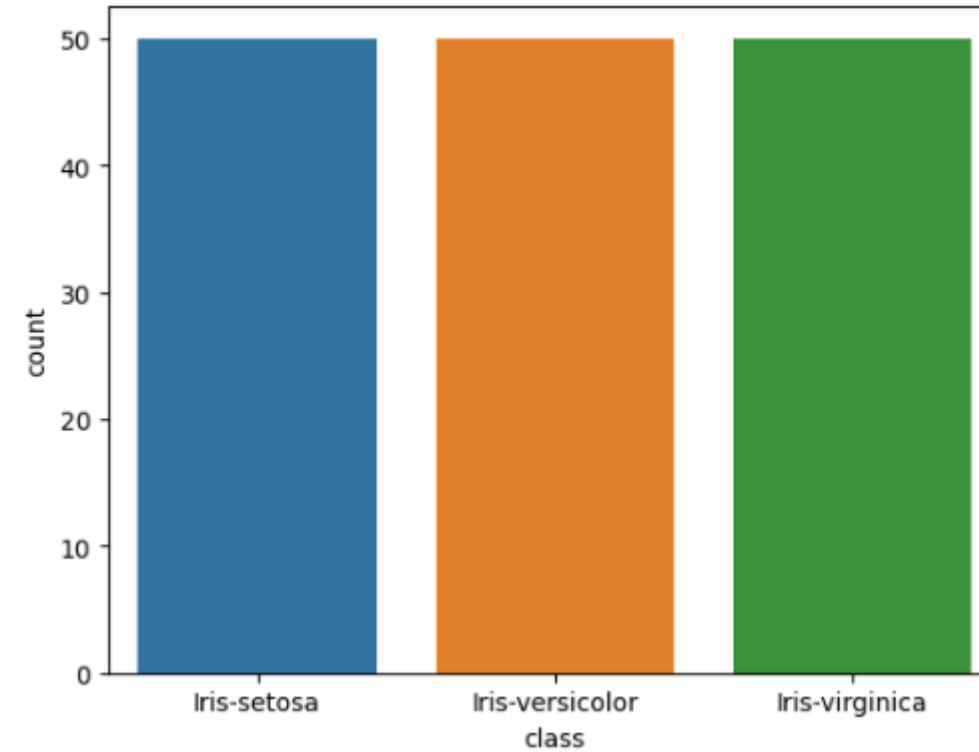
	sepalength	sepalwidth	petallength	petalwidth
sepalength	1.000000	-0.109369	0.871754	0.817954
sepalwidth	-0.109369	1.000000	-0.420516	-0.356544
petallength	0.871754	-0.420516	1.000000	0.962757
petalwidth	0.817954	-0.356544	0.962757	1.000000



# TP5 : Analyse du jeu de données Iris avec Matplotlib et Seaborn 1/5

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Clic droit + Enregistrer l'image sous
sns.countplot(x='class', hue='class', data=df)
plt.show()
```



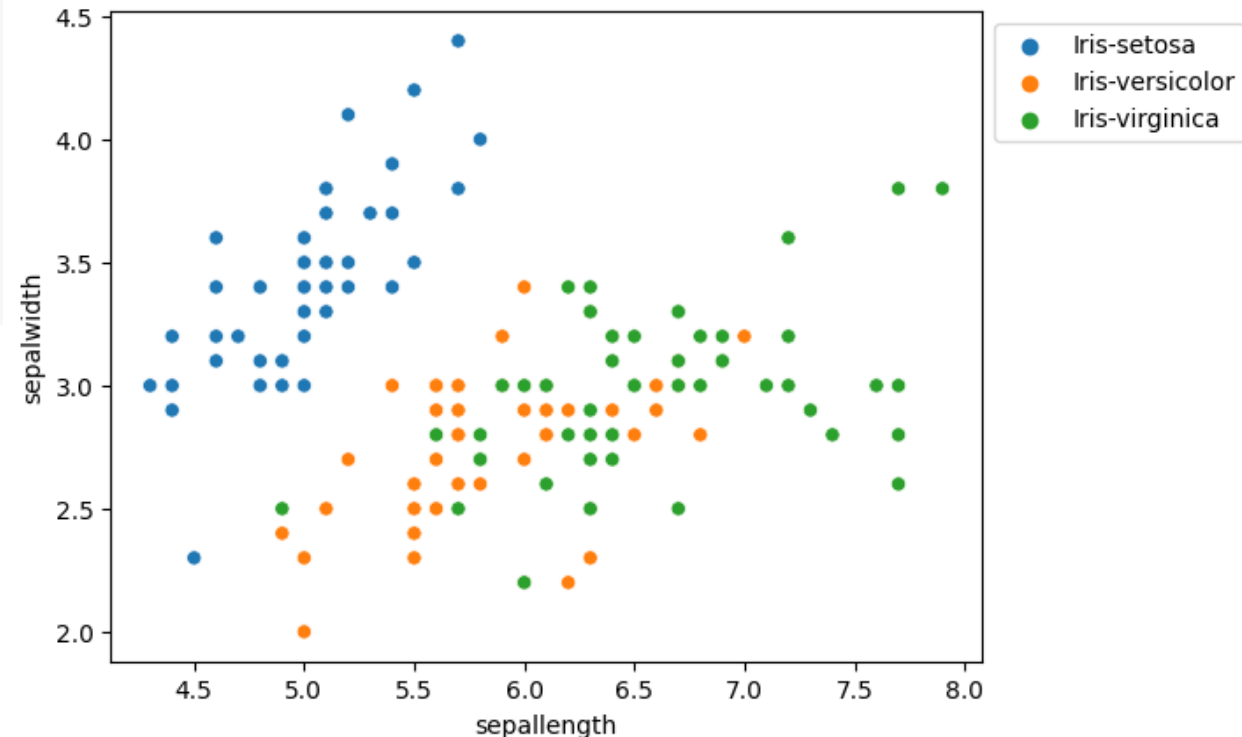
# TP5 : Analyse du jeu de données Iris avec Matplotlib et Seaborn 2/5

```
import seaborn as sns
import matplotlib.pyplot as plt

# Comparaison entre longueur et largeur sépales
sns.scatterplot(x='sepalength', y='sepalwidth', hue='class', data=df)

# Légende hors de la figure
plt.legend(bbox_to_anchor=(1, 1), loc=2)

plt.show()
```



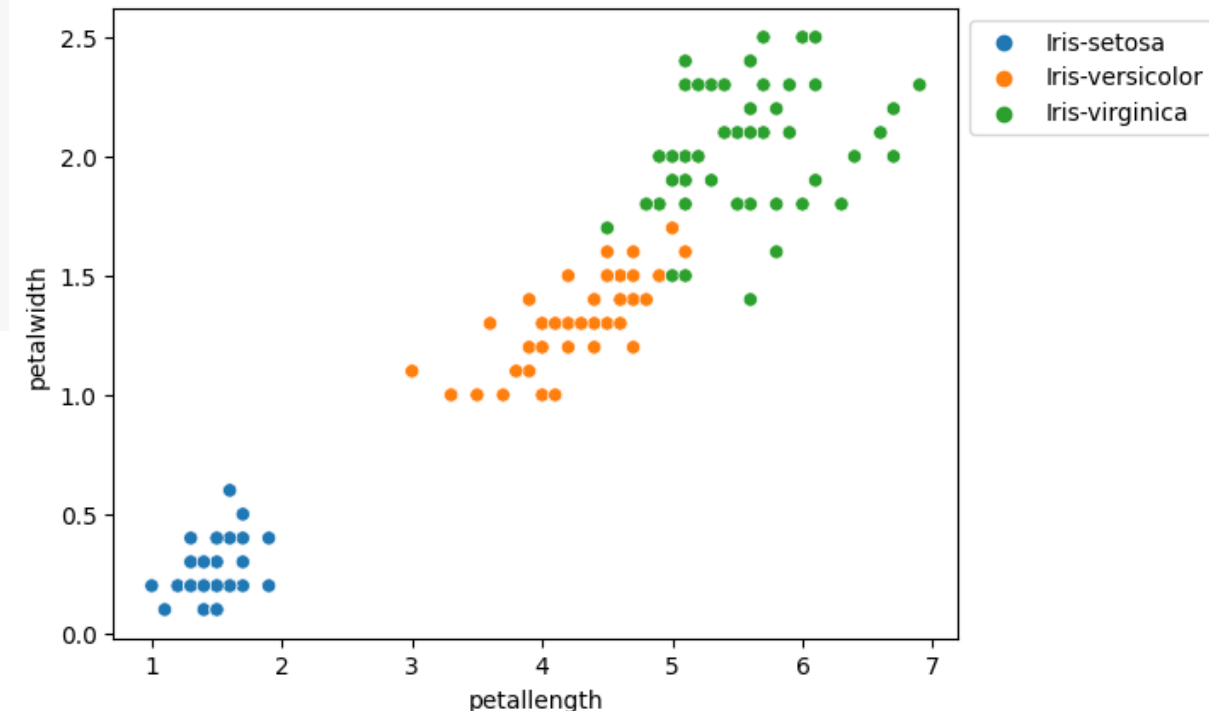
# TP5 : Analyse du jeu de données Iris avec Matplotlib et Seaborn 3/5

```
import seaborn as sns
import matplotlib.pyplot as plt

# Comparaison entre longueur et largeur pétales
sns.scatterplot(x='petallength', y='petalwidth', hue='class', data=df)

# Légende hors de la figure
plt.legend(bbox_to_anchor=(1, 1), loc=2)

plt.show()
```

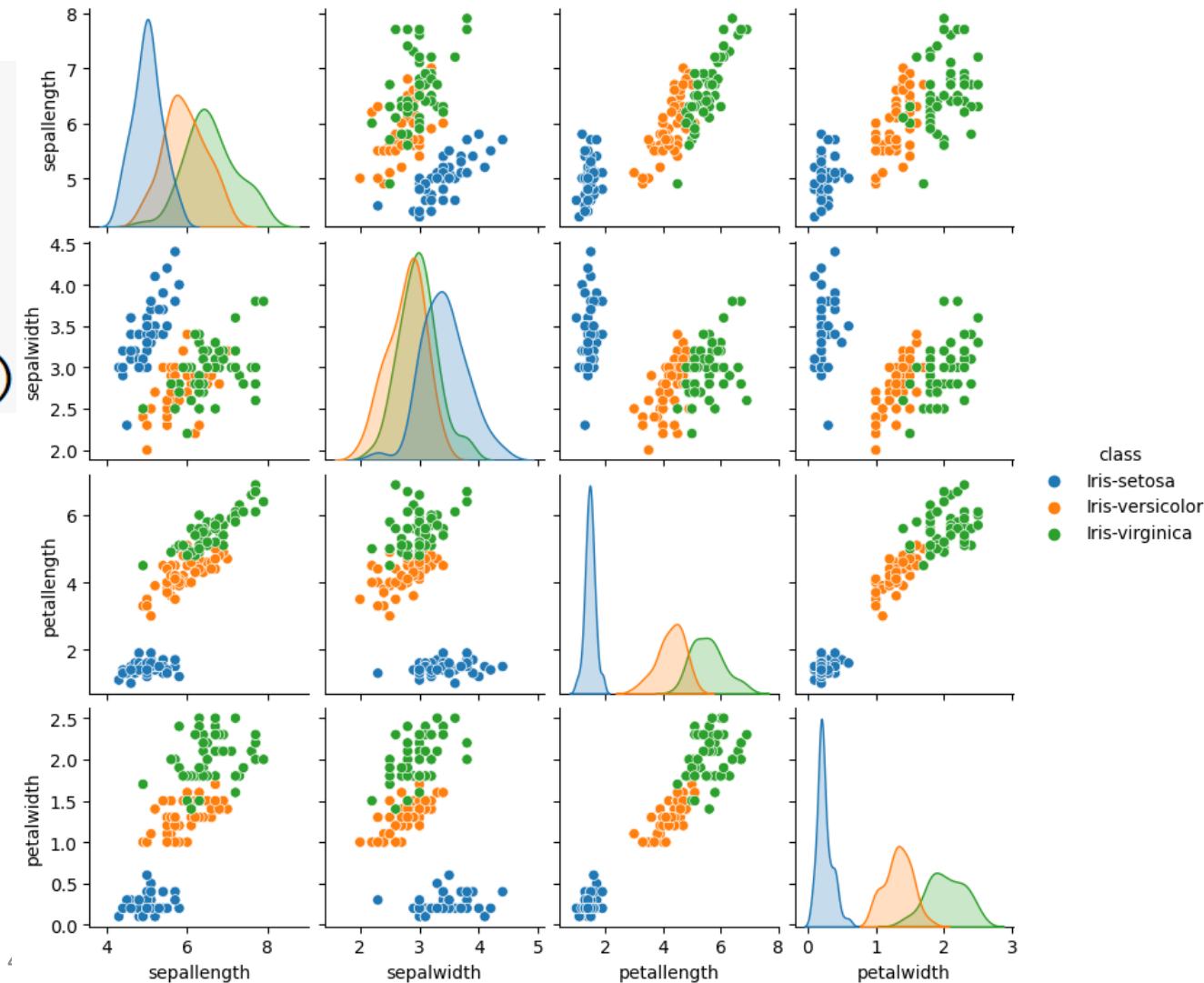


**PAUSE 15 mn**

# TP5 : Analyse du jeu de données Iris avec Matplotlib et Seaborn 4/5

```
import seaborn as sns
import matplotlib.pyplot as plt

# relation entre colonnes
sns.pairplot(df, hue='class', height=2)
```



# TP5 : Analyse du jeu de données Iris avec Matplotlib et Seaborn 5/5

## Histogramme

```
import seaborn as sns
import matplotlib.pyplot as plt

fig, axes = plt.subplots(2, 2, figsize=(10,10))

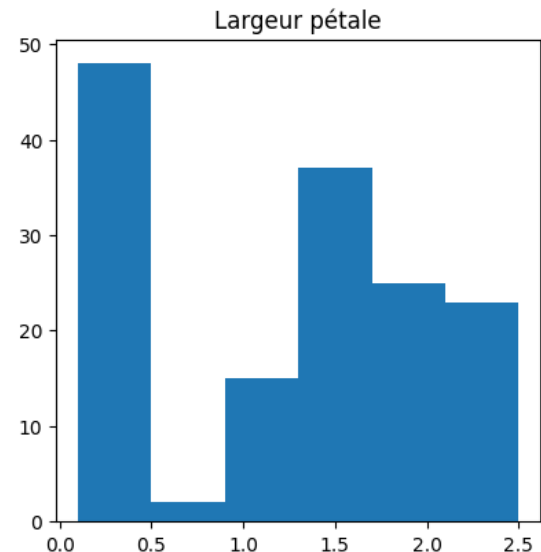
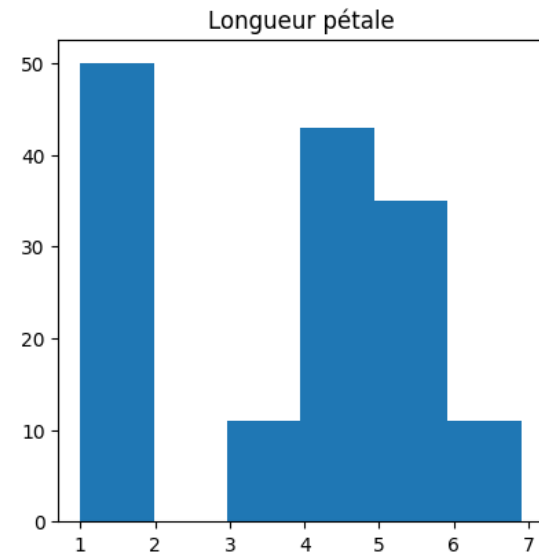
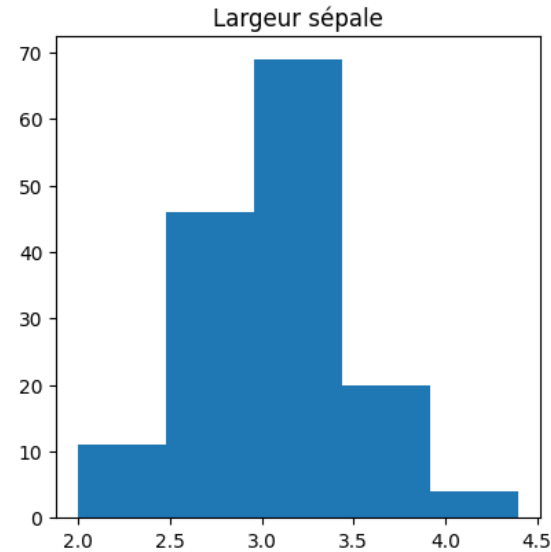
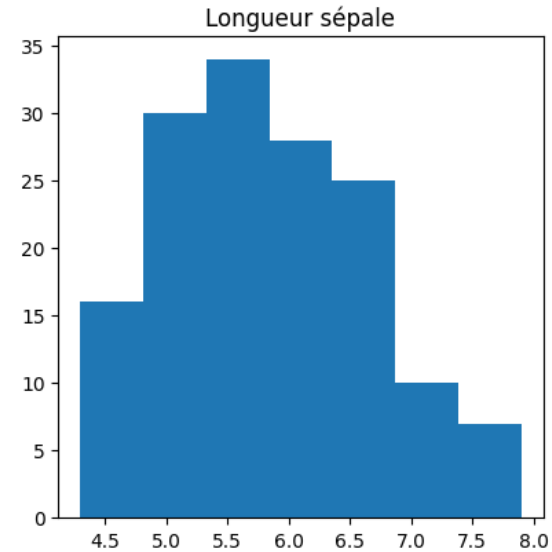
axes[0,0].set_title("Longueur sépale")
axes[0,0].hist(df['sepalength'], bins=7)

axes[0,1].set_title("Largeur sépale")
axes[0,1].hist(df['sepalwidth'], bins=5);

axes[1,0].set_title("Longueur pétale")
axes[1,0].hist(df['petallength'], bins=6);

axes[1,1].set_title("Largeur pétale")
axes[1,1].hist(df['petalwidth'], bins=6);
```

46



# Ressources pour aller plus loin

- **Apprenez les bases du langage Python**
  - <https://openclassrooms.com/fr/courses/7168871-apprenez-les-bases-du-langage-python>
- **Exploratory Data Analysis on Iris Dataset**
  - <https://www.geeksforgeeks.org/exploratory-data-analysis-on-iris-dataset/?ref=rp>
- **Analyse des données avec Python**
  - <https://www.coursera.org/learn/data-analysis-with-python>
- **EcoPy - Ecological Data Analysis in Python**
  - <https://ecopy.readthedocs.io/en/latest/>





**FIN DE LA FORMATION**

**MERCI**