Université de Monastir
Institut Supérieur d'Informatique et de Mathématiques de Monastir
Département d'Informatique

PROGRAMMATION PYTHON

Première Année Préparatoire Intégré
2025-2026

Séance 01

Présentation du langage Python

- Python est un langage de programmation créé en 1989 par Guido van Rossum, aux Pays-Bas.
- La première version publique de ce langage a été publiée en 1991.
- La dernière version de Python est la version 3. Plus précisément, la version 3.14 a été publiée en octobre 2025.
- La Python Software Foundation (https://www.python.org/psf/) est l'association qui organise le développement de Python et anime la communauté de développeurs et d'utilisateurs.
- Python est un langage simple, lisible et facilement maintenable, avec une syntaxe claire qui favorise la rapidité de développement.

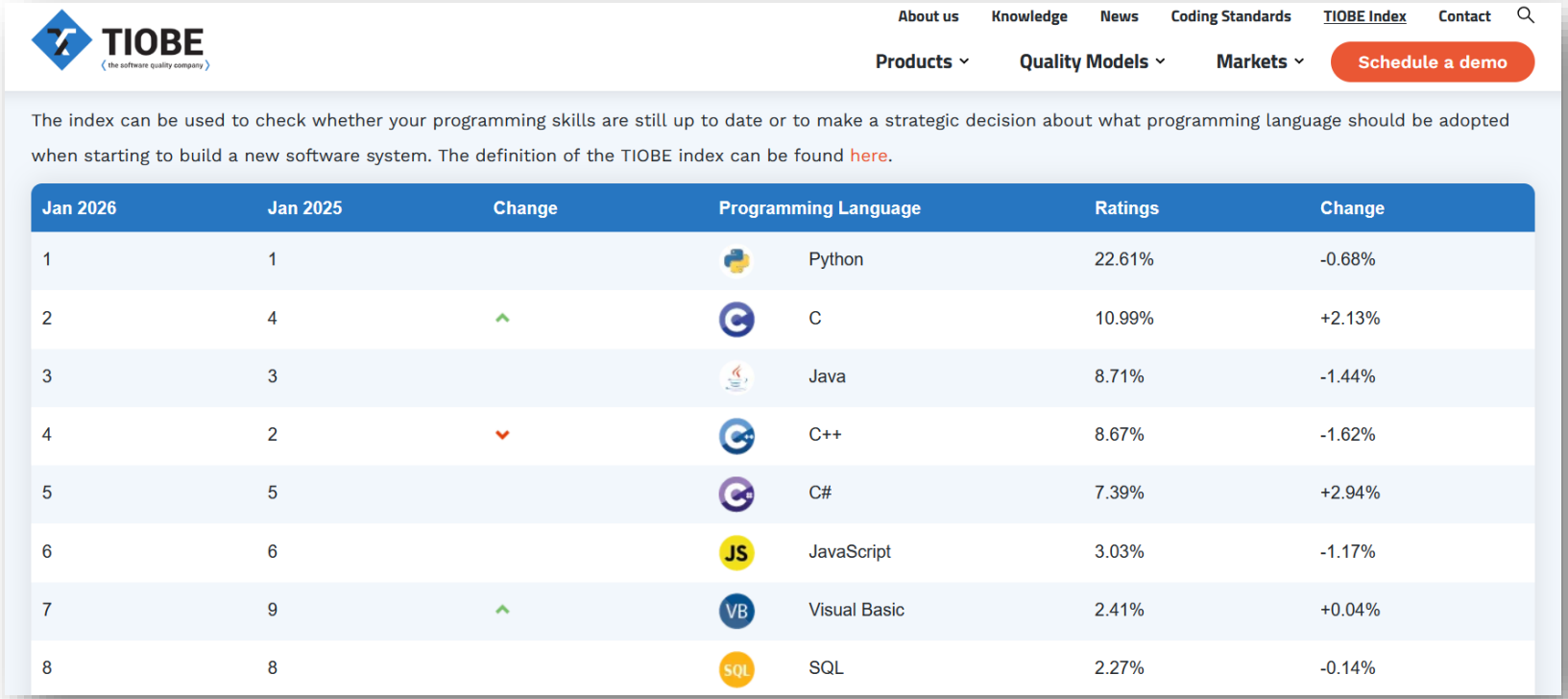
Caractéristiques principales du langage Python

Python présente de nombreuses caractéristiques intéressantes :

- Gratuit.
- Lisible et simple : indentation obligatoire pour structurer le code.
- Interprété : le code s'exécute directement, pas besoin de compilation.
- Multiplateforme : fonctionne sur Windows, Mac OS, Linux, Android, iOS, …
- Polyvalent : utilisé dans plusieurs domaines.
- Extensible : MAJ régulières des différentes bibliothèques et modules externes.
- Orienté objet : supporte la programmation objet et fonctionnelle.

Caractéristiques principales du langage Python

- Le langage le plus utilisé au monde (classements TIOBE).



TIOBE
the software quality company

About us | Knowledge | News | Coding Standards | TIOBE Index | Contact

Products | Quality Models | Markets | Schedule a demo

The index can be used to check whether your programming skills are still up to date or to make a strategic decision about what programming language should be adopted when starting to build a new software system. The definition of the TIOBE index can be found here.

Jan 2026	Jan 2025	Change	Programming Language	Ratings	Change
1	1		Python	22.61%	-0.68%
2	4	↑	C	10.99%	+2.13%
3	3		Java	8.71%	-1.44%
4	2	↓	C++	8.67%	-1.62%
5	5		C#	7.39%	+2.94%
6	6		JavaScript	3.03%	-1.17%
7	9	↑	Visual Basic	2.41%	+0.04%
8	8		SQL	2.27%	-0.14%

Utilisations courantes du langage Python

- **Développement Web :** Création de sites web dynamiques et d'applications côté serveur avec des frameworks comme **Django** et **Flask**.
- **IA :** Apprentissage automatique et approfondi (Machine Learning et Deep Learning).
- **Automatisation :** Automatisation de tâches répétitives (envoi d'emails, gestion de fichiers, …).
- **Développement de logiciels :** Création d'applications de bureau, de logiciels scientifiques, et même de jeux.
- **Big Data, Business Intelligence, Data Science :** Collecte et stockage massive de données, Analyse, visualisation et traitement de grands volumes de donnée, Prise de décision.
- **Robotique et Vision par Ordinateur :** Développement de systèmes intelligents et traitement d'images.

Université de Monastir
Institut Supérieur d'Informatique et de Mathématiques de Monastir
Département d'Informatique

TRAITEMENT ET ANALYSE DE DONNÉES AVEC PANDAS

pandas

Première Année Préparatoire Intégré
2025-2026

1 Introduction

Notion de Big Data

Traitement des données

Présentation de la bibliothèque pandas

?

Combien de données produit un smartphone par jour ?

En 2025, un smartphone génère ou consomme environ 20-22 Go de données par mois, soit environ 0,7-0,8 Go par jour.

?

Comment Google Maps prédit le trafic ?

Google Maps prédit le trafic en analysant en temps réel les vitesses de millions d'utilisateurs, combinées à l'historique et à l'intelligence artificielle, afin d'anticiper l'état futur des routes.

Comment Netflix recommande des films ?

Netflix recommande des films en analysant le comportement de millions d'utilisateurs grâce au Big Data et au machine learning afin de prédire ce que chacun est le plus susceptible d'aimer.

Interprétation

- Chaque jour, des milliards de messages sont envoyés, des millions de vidéos sont visionnées, des paiements sont effectués et des capteurs collectent des données en continu.

- Smartphones, réseaux sociaux, objets connectés, plateformes numériques et systèmes intelligents produisent une quantité massive d’informations à une vitesse jamais atteinte auparavant.

Interprétation

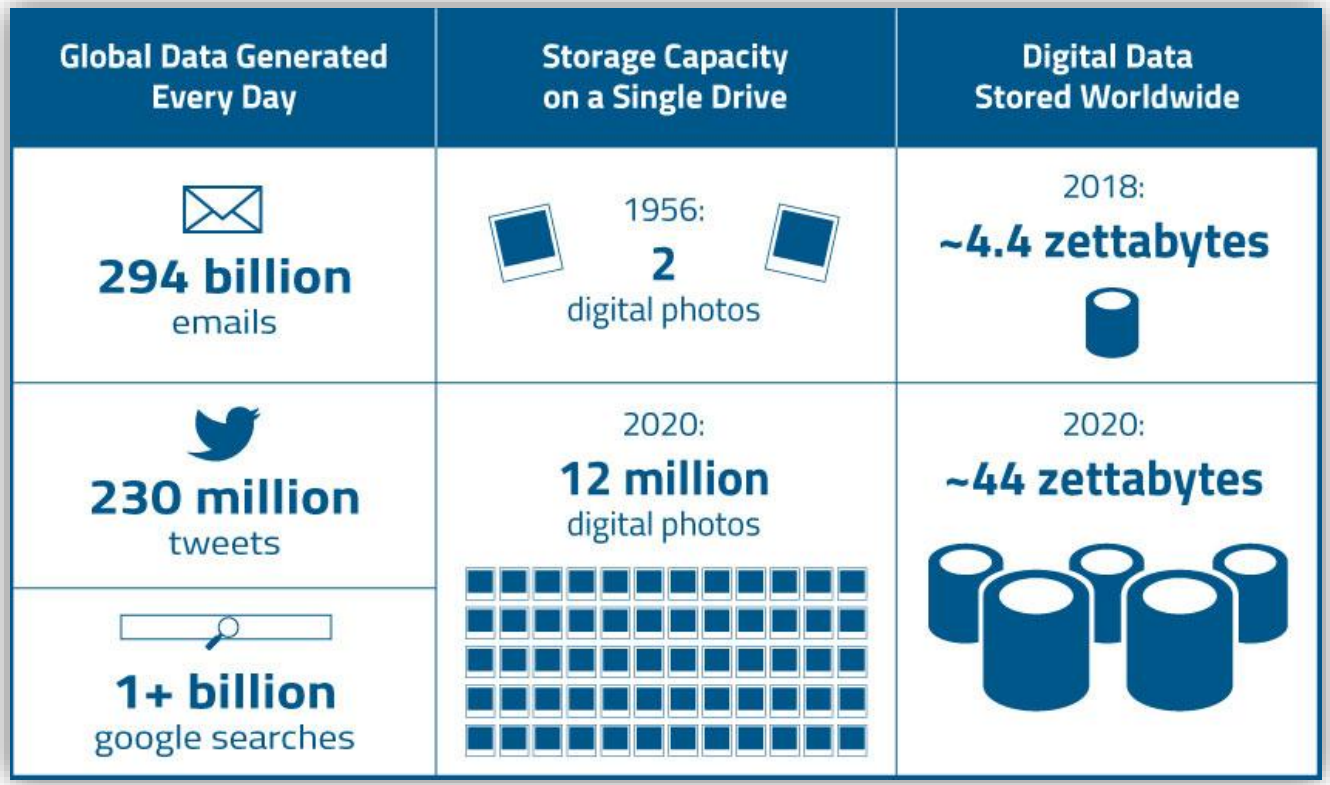
- Face à cette explosion des données, les outils informatiques classiques montrent leurs limites. Il devient difficile, voire impossible, de stocker, gérer et analyser efficacement ces volumes gigantesques d'informations à l'aide d'un simple ordinateur ou d'une base de données traditionnelle.
- Cette transformation a donné naissance à un nouveau domaine : le **Big Data**.

1 Définition : Big Data

- Le **Big Data** désigne l'ensemble des technologies et des méthodes permettant de stocker, traiter et analyser des données massives, hétérogènes et générées à grande vitesse, afin d'en extraire de la valeur et d'aider à la prise de décision.

- Le Big Data est aujourd'hui au cœur de nombreux secteurs : santé, finance, transport, éducation, industrie, cybersécurité et intelligence artificielle.

1 Croissance du stockage digital



https://newsroom.lamresearch.com/image/Lam_Blog_Data_Storage.jpg

1 Kilooctet (Ko) = 2^{10} octets

1 Mégaoctet (Mo) = 2^{10} Ko

1 Gigaoctet (Go) = 2^{10} Mo

1 Téraoctet (To) = 2^{10} Go

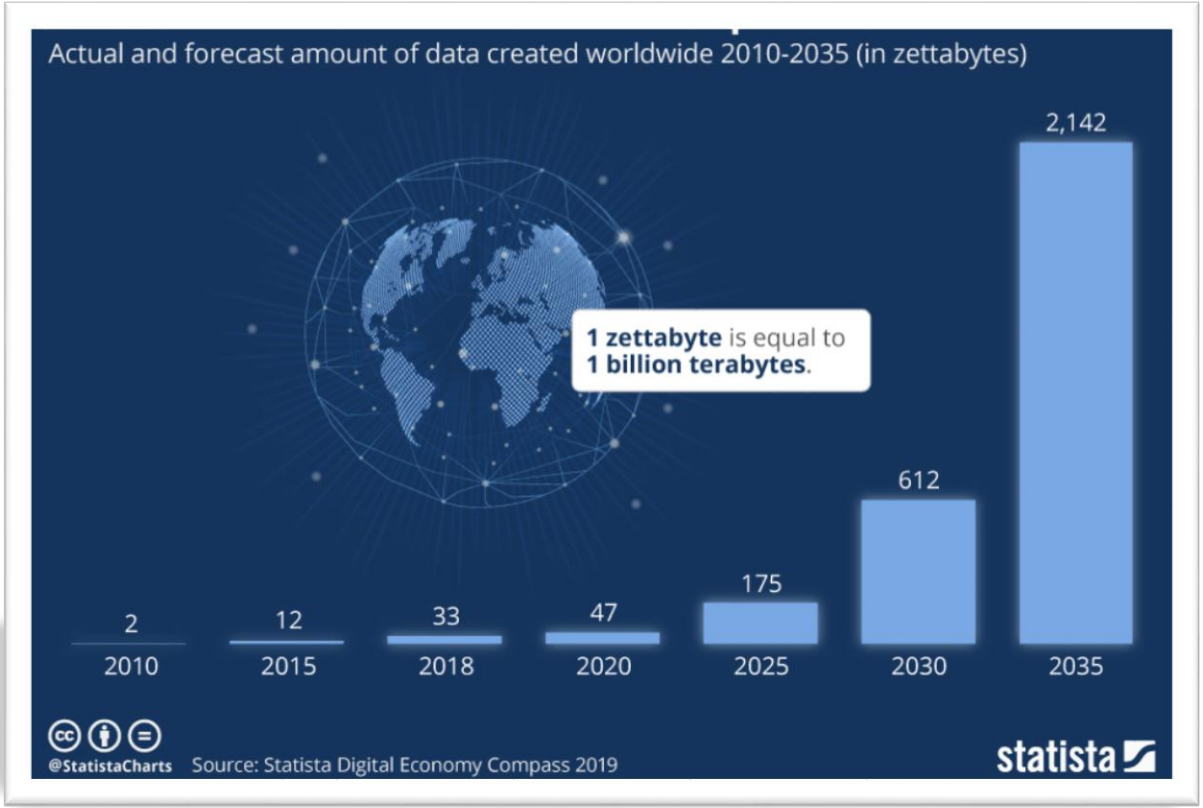
1 Pétaoctet (Po) = 2^{10} To

1 Exaoctet (Eo) = 2^{10} Po

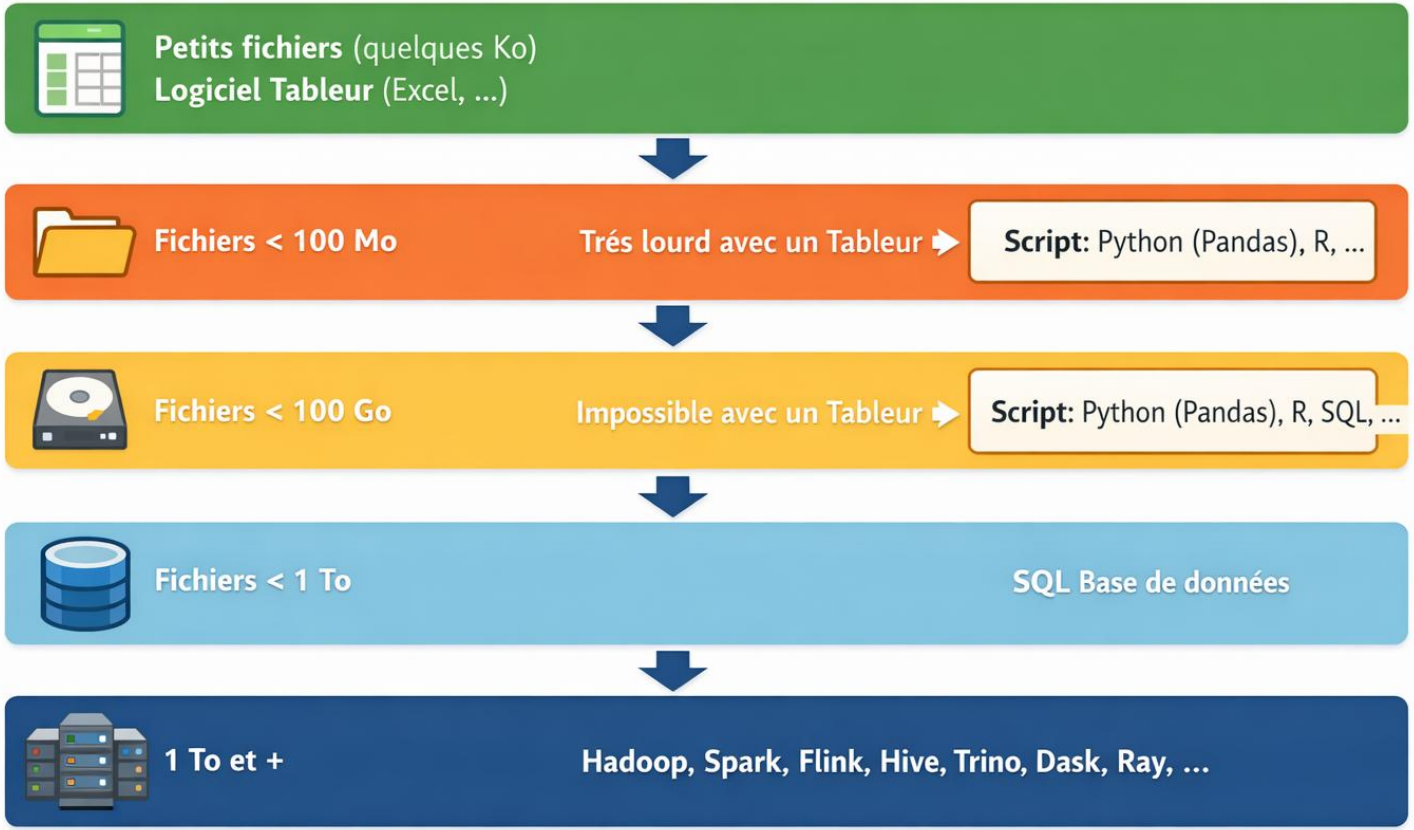
1 Zettaoctet (Zo) = 2^{10} Eo

1 Yottaoctet (Yo) = 2^{10} Zo

1 Croissance du stockage digital



1 Les bons outils pour traiter les données



1 Les bons outils pour traiter les données

- L’utilisation des anciennes méthodes basées sur les logiciels Tableur (Excel) est simple avec des données légères mais délicate pour des traitements complexes (VBA).

- L’utilisation des nouvelles méthodes basées sur les langages de programmation (Python) permet des traitements plus complexes (nettoyage, analyse, …) et offre des performances nettement meilleures qui s’adaptent au Big Data.

1 Pandas (Panel Data Analysis), c’est quoi ?

- Une des bibliothèques Python les plus utilisées pour analyser les données
- Développé en 2008 par Wes McKinney
- Open source
- Implémenté à partir de C – d’où sa rapidité
- A introduit des structures de données spécifiques telles que : **Series**, **DataFrame**, **Pannel**.
- Offre une productivité et des performances élevées aux utilisateurs

1 Principales caractéristiques

- Gestion facile des données manquantes.
- Mutabilité de taille : les données peuvent être insérées et supprimées aisément.
- Application d'indices automatiques et utilisation d'étiquettes personnalisées.
- Découpage intelligent en sous-ensembles des données de grandes tailles.
- Fusion et jonction intuitives d'ensembles de données.
- Traitements croisés dynamiques (pivotement) des données.
- Outils d'E/S robustes pour charger des données à partir de fichiers textes, de fichiers Excel, de bases de données, …

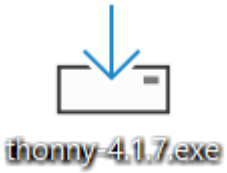
2 Installation

Installation de l’environnements de travail

Installation des bibliothèques

2 Installation de l'environnement de travail

- **Thonny** est un **IDE** (Integrated Development Environment) Python pédagogique pour débutants, simple à utiliser et avec débogage pas à pas.
- Télécharger puis installer Thonny à partir de l'adresse : **https://thonny.org/**

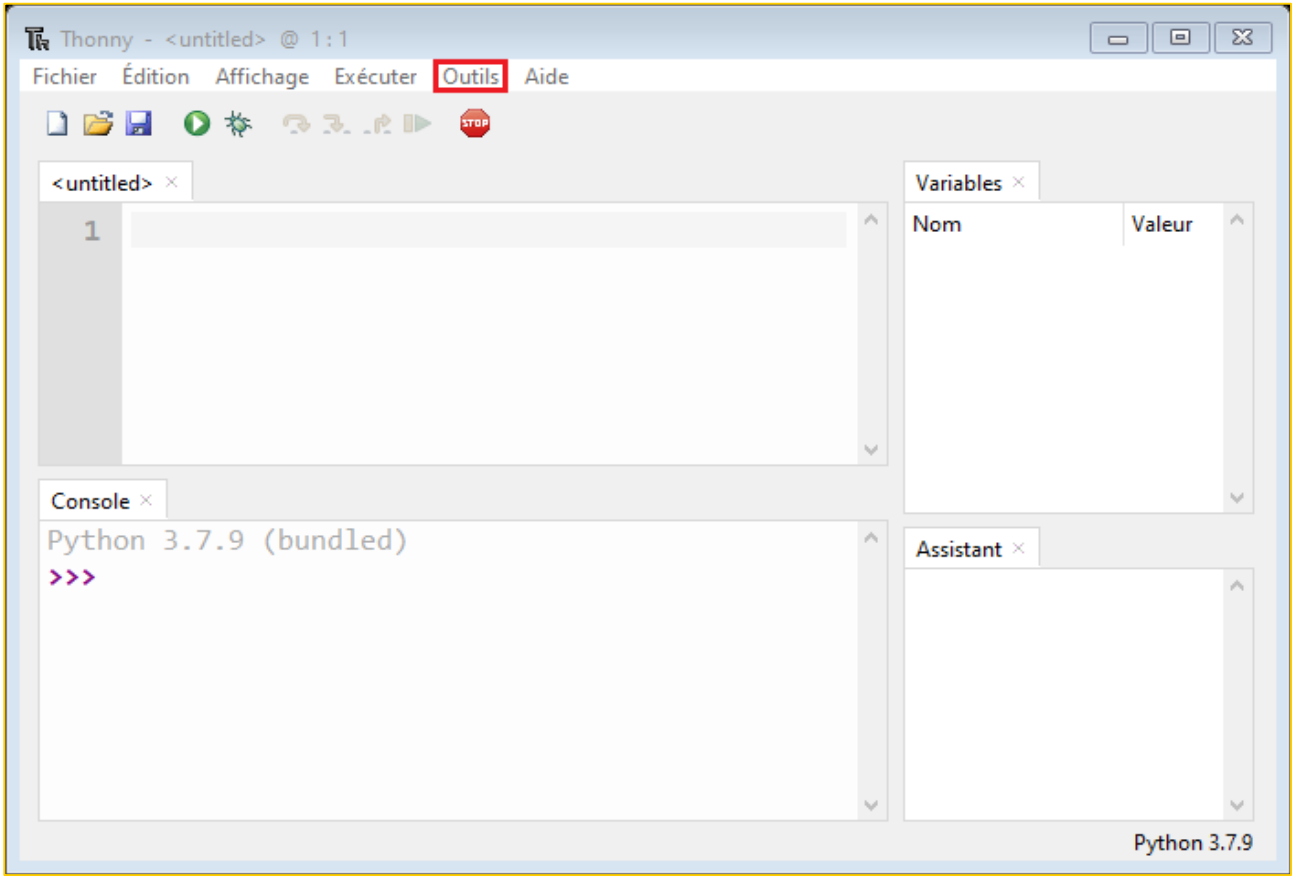


2 Présentation de l'interface

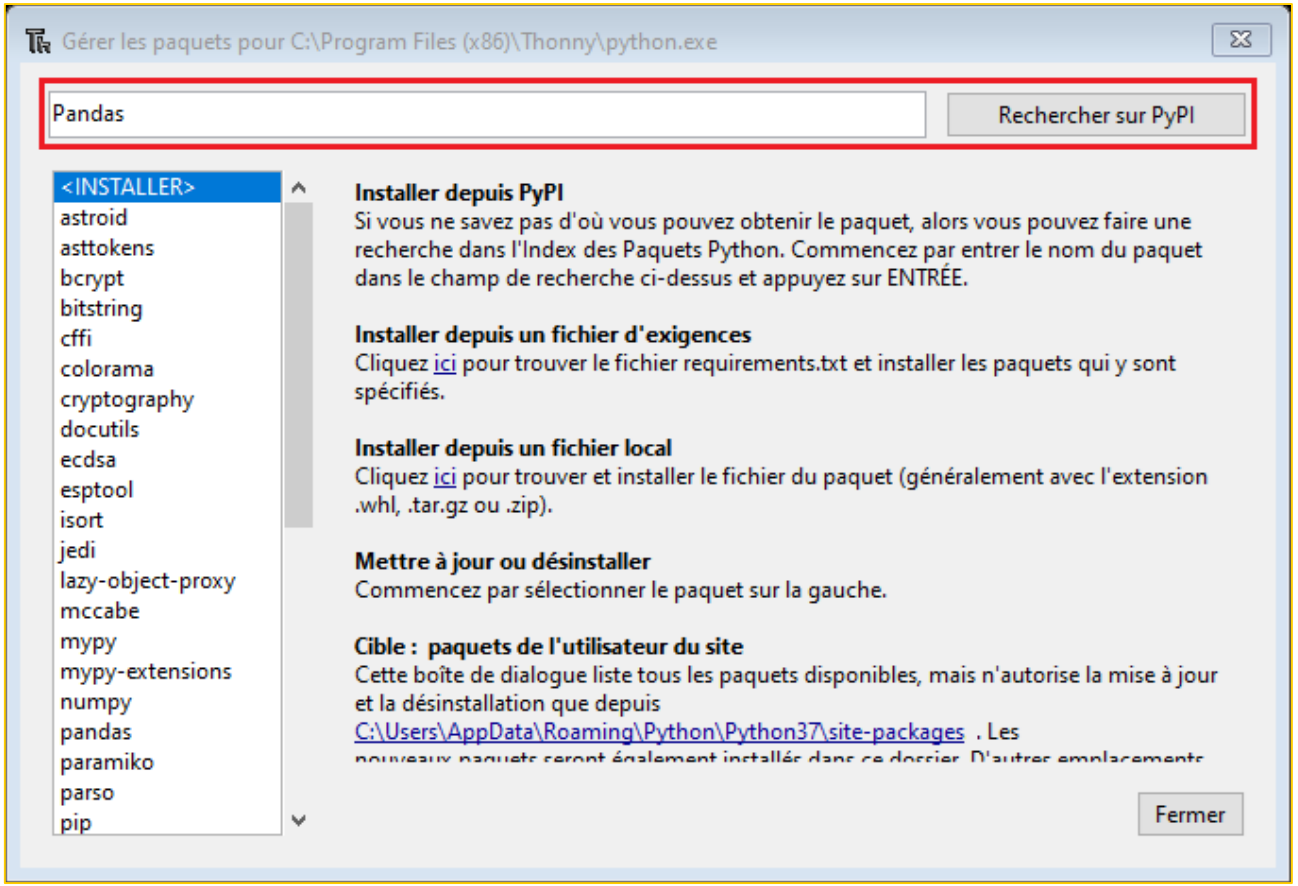
2 Installation des bibliothèques

- Avec "**Thonny**", l'installation d'une bibliothèque se fait comme suit :
  - Cliquer le menu "**Outils**" puis la commande "**Gérer les paquets**"
  - Dans la zone de recherche, taper le nom de la bibliothèque ("**pandas**", "**numpy**", "**openpyxl**", "**matplotlib**") puis cliquer "**Rechercher sur PyPI**"
  - Sélectionner le nom de la bibliothèque puis l'installer.

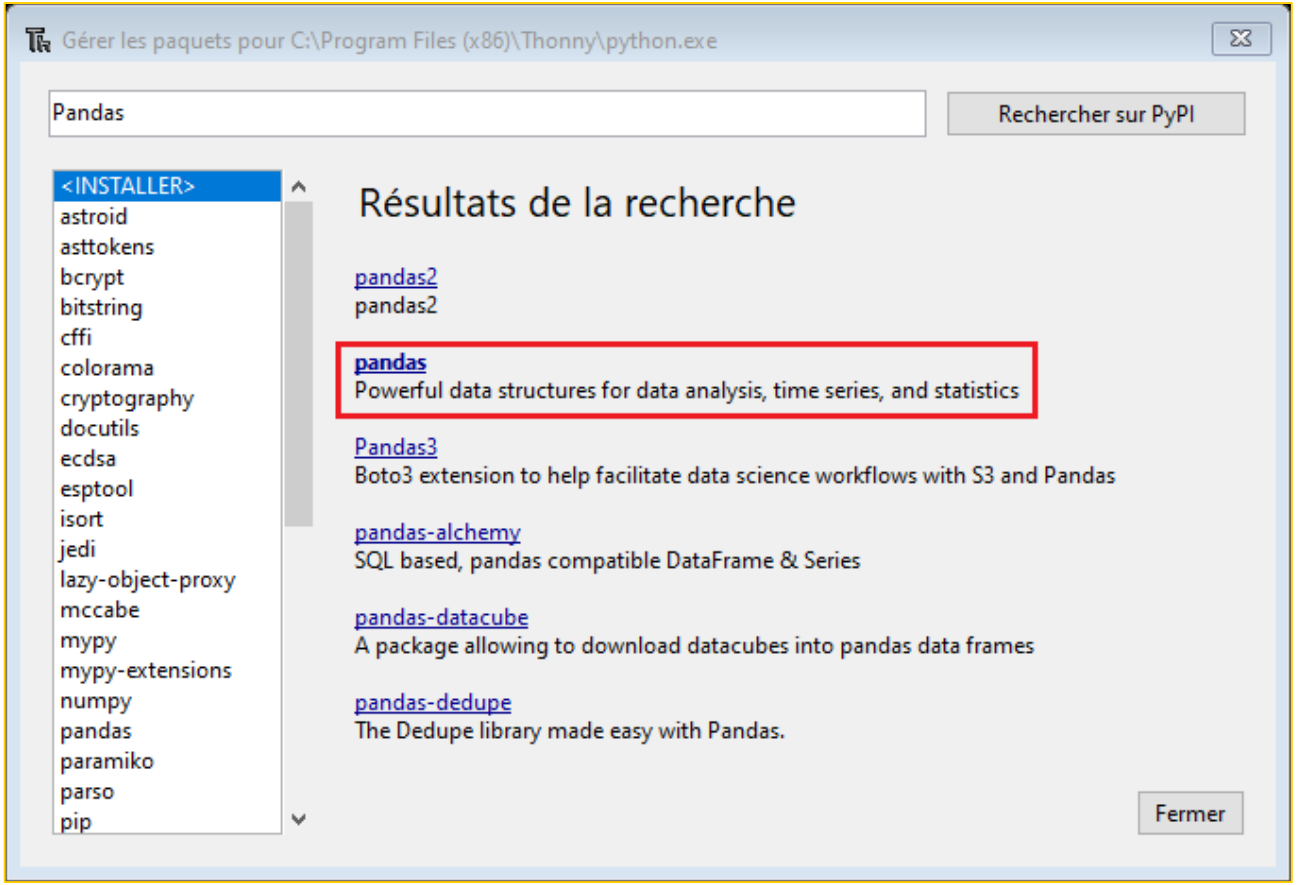
2 Installation des bibliothèques



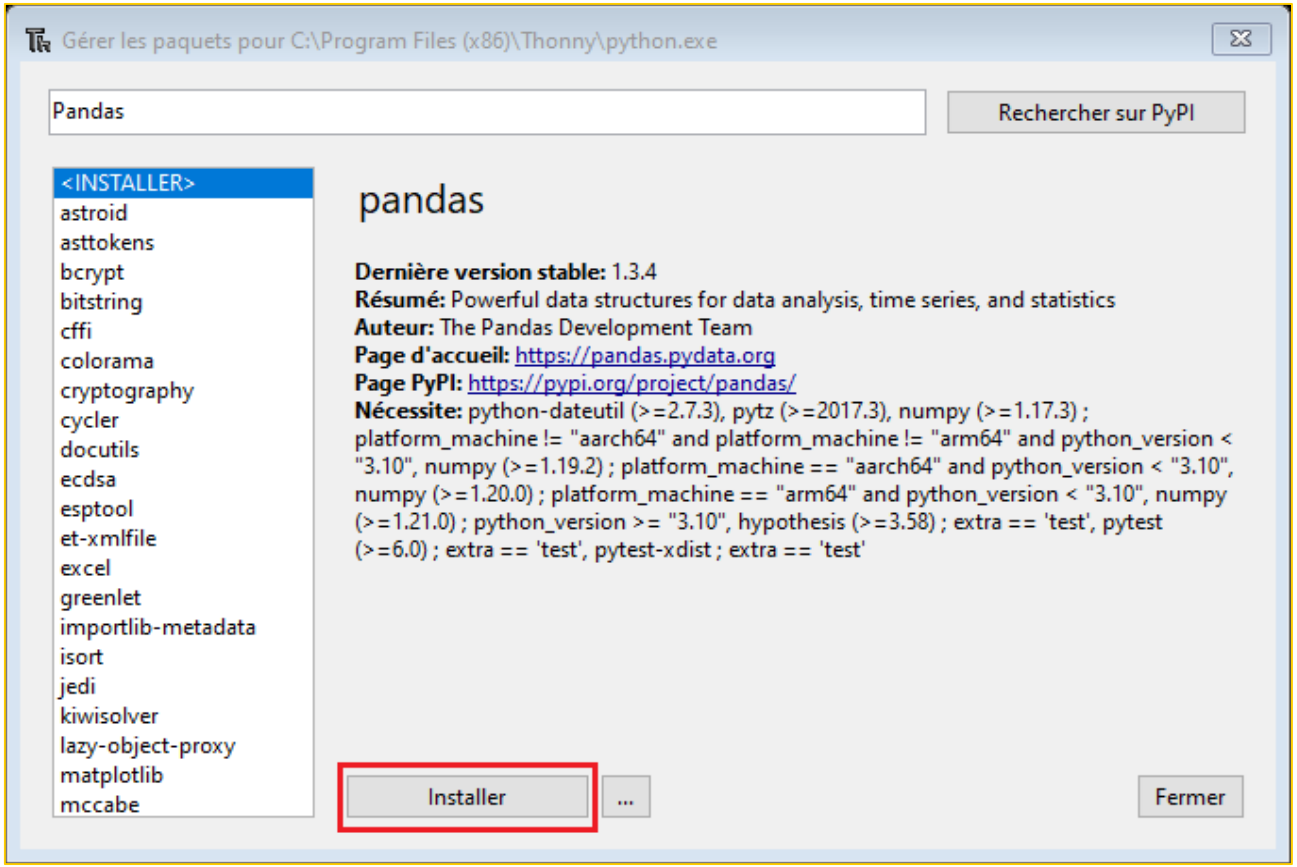
2 Installation des bibliothèques



2 Installation des bibliothèques



2 Installation des bibliothèques



3 Importation des bibliothèques

Importation de la bibliothèque pandas

Importation de la bibliothèque numpy

Importation de la bibliothèque matplotlib

3 Comment je l'importe ?

- Importation de toute la bibliothèque :
  - `import nom_bibliothèque`
- Importation avec un alias (nom abrégé) :
  - `import nom_bibliothèque as alias`
- Importation d'un élément précis d'une bibliothèque :
  - `from nom_bibliothèque import élément`
- Importation de plusieurs éléments :
  - `from nom_bibliothèque import élément1, élément2`
- Importation de tous les éléments :
  - `from nom_bibliothèque import *`

3 Comment je l'importe ?

- **Pandas :**

```
import pandas as alias
import pandas as ps
```

- **Numpy :**

```
import numpy as alias
import numpy as np
```

- **Matplotlib :**

```
import matplotlib.pyplot as alias
import matplotlib.pyplot as plt
```

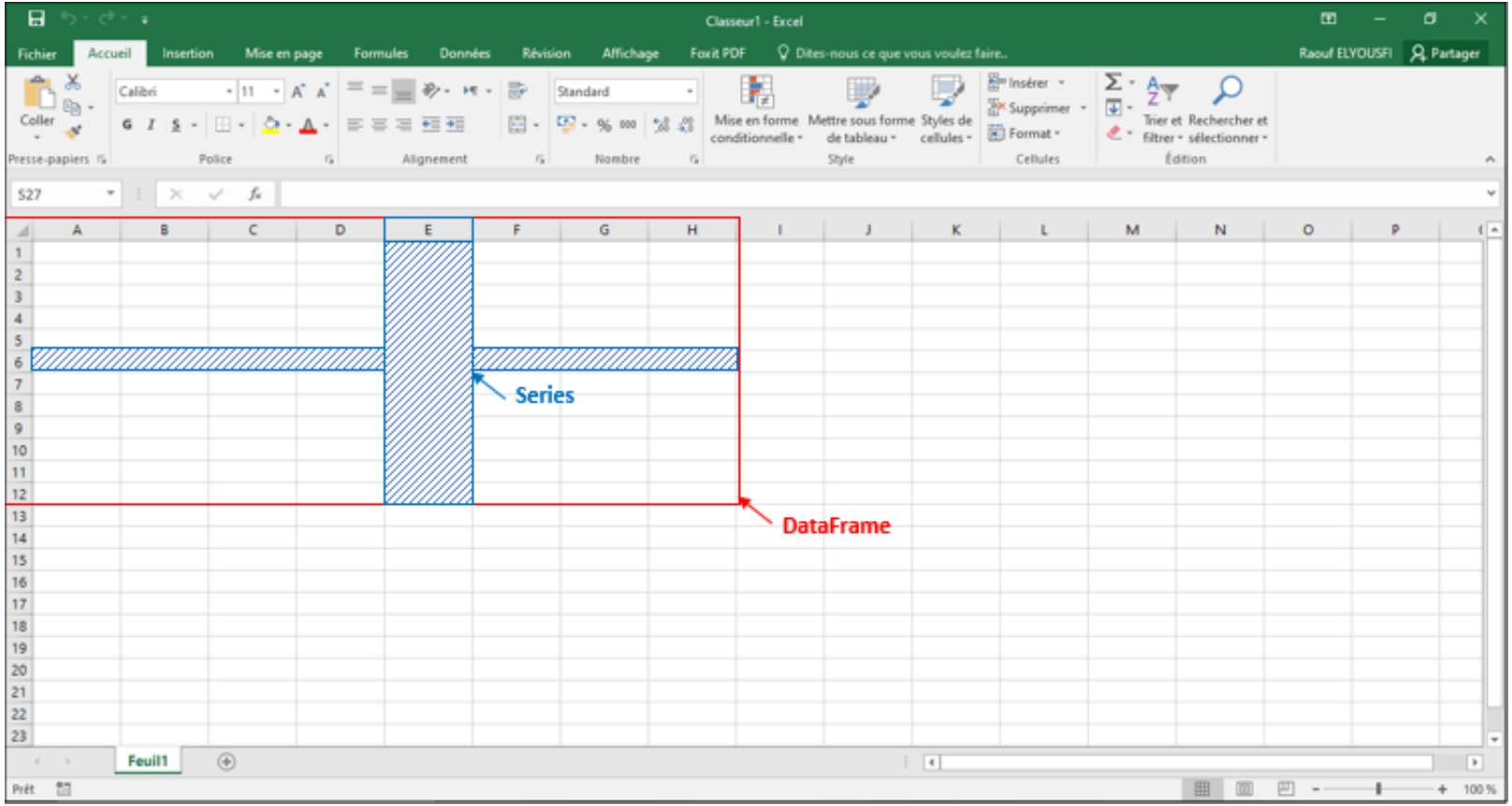
C'est **la convention universelle** en data science.

4 Les structures de données en Pandas

La structure de données "Series"

La structure de données "DataFrame"

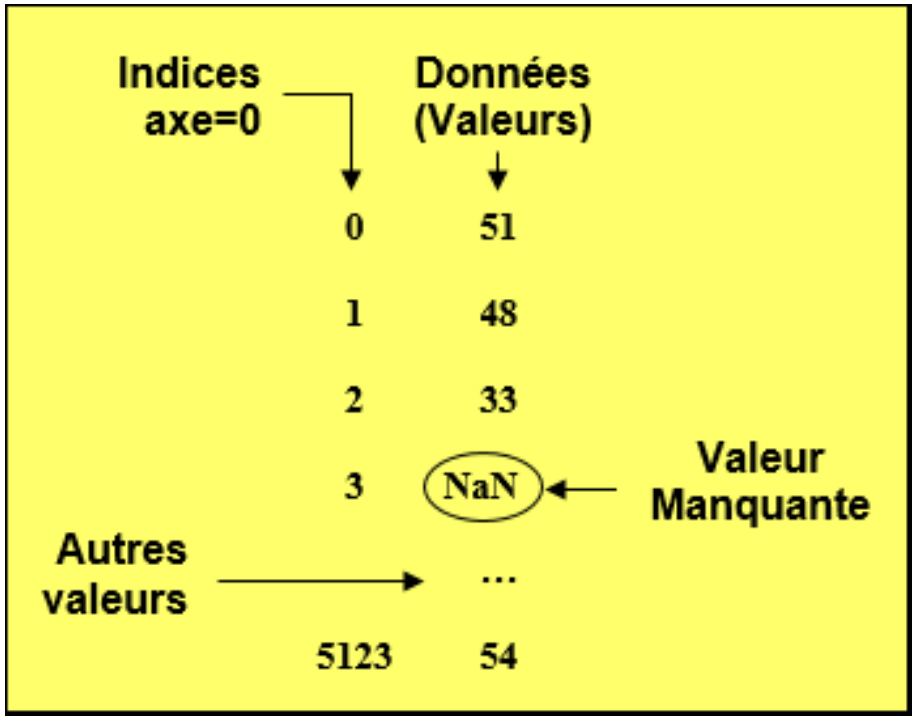
4 Quelles ressemblances avec une feuille Excel ?



4 Qu’est-ce qu’un objet "Series" ?

- Un objet **Series** correspond à un vecteur (tableau) à 1 dimension.
- Un objet **Series** est un objet de données mutable ➔ Mettre à jour le contenu + Ajouter et/ou Supprimer des éléments.
- La taille d’un objet **Series** est immutable ➔ Toute opération d’ajout ou de suppression crée en réalité une nouvelle **Series**.
- Il peut contenir des données de tout type.
- Un objet **Series** peut être considéré comme une colonne verticale pouvant contenir plusieurs lignes ou une ligne horizontale pouvant contenir plusieurs colonnes.

4 Qu’est-ce qu’un objet "Series" ?



4 Comment créer un objet Series ?

#Création d'une Series avec des indices par défaut

Id_Serie = alias.Series([val1, …, valN])

```
import pandas as ps
s1 = ps.Series(['Amine', 10, 'Monastir'])
print(s1)
```

```
0        Amine
1           10
2     Monastir
dtype: object
```

4 Comment créer un objet Series ?

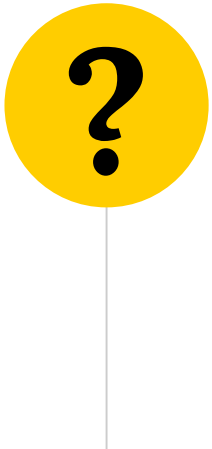
#Création d'une Series avec des indices personnalisés

Id_Serie =alias.Series([Val1, ..., ValN], index=[Idx1, ..., IdxN])

```
import pandas as ps
s2=ps.Series(['Amine', 10, 'Monastir'], index=['Prénom', 'Age', 'Ville'])
print(s2)
```

```
Prénom        Amine
Age              10
Ville      Monastir
dtype: object
```

Que se passe-t-il si l'une des valeurs est utilisée comme un nom d'indice ?



Remarque

- Il est possible de créer une série vide de la façon suivante :

```
import pandas as alias
Id_Serie = alias.Series()
```

```
import pandas as ps
s0=ps.Series()
print(s0)
```

```
Series([], dtype: float64)
```

4 Comment afficher les données d'une Series ?

```
#Tout le contenu d'une série
print(Id_Serie)
```

print(s1)

```
0        Amine
1           10
2    Monastir
dtype: object
```

print(s2)

```
Prénom        Amine
Age              10
Ville     Monastir
dtype: object
```

4 Comment afficher les données d'une Series ?

#Une seule valeur de la série

print(Id_Serie[Indice]) *ou bien* **print(Id_Serie[[Indice]])**

print(s1[0])

```
Amine
```

print(s2['Ville'])

```
Monastir
```

print(s1[[0]])

```
0     Amine
```

print(s2[['Ville']])

```
Ville      Monastir
```

4 Comment afficher les données d'une Series ?

```
#Valeurs multiples non successives
print(Id_Serie[ [Indice1,Indice2, ...] ])
```

```
print(s1[[0,2]])
```

```
0       Amine
2    Monastir
dtype: object
```

```
print(s2[['Prénom','Ville']])
```

```
Prénom       Amine
Ville     Monastir
dtype: object
```

4 Comment afficher les données d'une Series ?

```
#Valeurs multiples successives
print(Id_Serie[Indice1 : Indice2])
```

```
print(s1[0:2])
```

```
0       Amine
1          10
dtype: object
```

```
print(s2['Prénom':'Ville'])
```

```
Prénom          Amine
Age                10
Ville      Monastir
dtype: object
```

4 Comment modifier les données d'une Series ?

Id_Serie [Indice] = Valeur *ou bien*

Id_Serie [[Indice1, …, IndiceN]] = [Valeur1, …, ValeurN]

```
s3=ps.Series([10, 20, 30, 40], index = ['a', 'b', 'c', 'd'])
s3['c']=300
s3['z']=50
print(s3)
```

```
a      10
b      20
c     300
d      40
z      50
dtype: int64
```

4 Comment filtrer les données d'une Series ?

```
print(Id_Serie[Condition])
```

```
print(s3[s3>40])
```

```
c     300
z      50
dtype: int64
```

4 Comment filtrer les données d'une Series ?

print(Id_Serie[Condition(s)])

On peut combiner plusieurs critères de sélection avec les opérateurs logiques & (pour **ET**) et | (pour **OU**)

print(s3[(s3<15) | (s3>150)])

```
a      10
c     300
dtype: int64
```

print(s3[(s3>20) & (s3<100)])

```
d     40
z     50
dtype: int64
```