



Séance d'orientation · Filières Développement informatique et Data

Prof. SABRI TARIK – sabritarik@hotmail.com

Programme de la séance



1

Pourquoi apprendre Python ?

2

Les 12 domaines d'application

3

Filière Développement informatique

4

Filière Data

5

Démarrer et réussir avec Python

6

Pourquoi vous lancer maintenant

01

Pourquoi apprendre Python ?

Python en bref



Créé en 1991

Conçu par Guido van Rossum, avec une idée simple : un code lisible comme de l'anglais.



Libre et gratuit

Open source : on peut l'utiliser, l'étudier et le partager sans licence.



Multi-paradigme

Programmation procédurale, orientée objet et fonctionnelle dans un seul langage.

Plus de 30 ans d'évolution



Une syntaxe proche du langage naturel



```
nom = input("Ton prénom ? ")
notes = [12, 15, 9, 17]
moyenne = sum(notes) / len(notes)

print(f"Bonjour {nom} !")
print(f"Ta moyenne : {moyenne:.1f}")
```

Lire, calculer et afficher en 5 lignes

Pourquoi c'est simple

Pas d'accolades ni de ;

L'indentation structure le code

Typage dynamique

Messages d'erreur explicites

Les 6 raisons d'apprendre Python

1

Facile et rapide à apprendre

2

Polyvalent : un seul langage, des dizaines de métiers

3

Communauté mondiale et documentation gratuite

4

Très demandé sur le marché de l'emploi

5

Productivité : moins de code, plus de résultats

6

Libre, gratuit et multiplateforme

Raison 1 : facile à apprendre

Le meilleur premier langage pour débiter en programmation



Syntaxe claire

On se concentre sur la logique,
pas sur la ponctuation.



Résultats rapides

Un premier programme utile
après quelques heures de
pratique.



Idéal pour débiter

Utilisé dans les écoles et
universités du monde entier.

Raison 2 : un langage polyvalent

Du site web à l'intelligence artificielle



Web

Sites et API.



Data

Analyse et visualisation.



IA

Machine et deep learning.



Automatisation

Scripts et outils.

Raison 3 : une communauté immense



Des millions de développeurs

Une réponse existe presque toujours à votre question.



Ressources gratuites

Documentation officielle, tutoriels, vidéos et cours en ligne.



Paquets installables

Des centaines de milliers de paquets installables avec une seule commande : pip.

Raison 4 : un atout pour l'emploi

Python figure parmi les compétences les plus recherchées



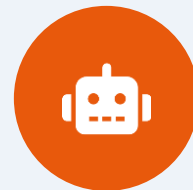
Développeur

Web, back-end, logiciels.



**Data analyst /
scientist**

Données et modèles.



Ingénieur IA

Machine learning et
LLM.



DevOps

Automatisation et cloud.

Raison 5 : productivité maximale



Prototypage rapide

Passer de l'idée à une démonstration en quelques heures.



Moins de code

Un programme Python est souvent 3 à 5 fois plus court qu'en Java.



Notebooks interactifs

Jupyter mélange code, résultats, graphiques et explications.

Raison 6 : libre et multiplateforme



Gratuit et libre

Aucune licence à payer, code source ouvert.



Windows, Linux, macOS

Le même programme fonctionne partout.



Adopté par les grands

Google, Netflix, la NASA et bien d'autres l'utilisent.

Un écosystème de bibliothèques

Il suffit de « pip install » pour gagner des mois de travail



Données

NumPy, pandas, Polars



Web

Django, Flask, FastAPI



IA

scikit-learn, PyTorch,
TensorFlow



Visualisation

Matplotlib, Seaborn,
Plotly

02

Les domaines d'application

Python : 12 domaines à explorer



Développement
web



Analyse de
données



Visualisation de
données



Machine learning



Deep learning et
vision



NLP et IA
générative



Automatisation et
scripts



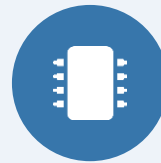
Systèmes, DevOps
et réseaux



Cybersécurité



Applications et
jeux



IoT et robotique



Finance et sciences

Domaine : Développement web

Créer sites, API et applications web robustes avec peu de code.



Sites et portails

Django fournit ORM, authentification et interface d'administration prêts à l'emploi.



API REST

Flask et FastAPI exposent des services pour applications mobiles et front-end.



E-commerce et CMS

Boutiques en ligne, blogs, plateformes de cours et tableaux de bord.

Développement web : exemple en Python



```
from fastapi import FastAPI
app = FastAPI()

@app.get("/etudiants/{id}")
def lire(id: int):
    return {"id": id, "nom": "Sara"}

# lancer : uvicorn main:app --reload
```

Bibliothèques clés

Django

Flask

FastAPI

Jinja2

Requests

SQLAlchemy

Une API web complète en quelques lignes

Domaine : Analyse de données

Transformer des fichiers bruts en informations exploitables.



Nettoyage

pandas gère valeurs manquantes, doublons et formats incohérents.



Exploration

Statistiques, regroupements, tableaux croisés et corrélations.



Rapports automatiques

Lire et produire des fichiers Excel et CSV sans effort manuel.

Analyse de données : exemple en Python



```
import pandas as pd

df = pd.read_csv("ventes.csv")
df = df.dropna()
top = (df.groupby("ville")["montant"]
        .sum().sort_values(ascending=False))
print(top.head(3))
```

Les 3 meilleures villes en 6 lignes

Bibliothèques clés

pandas

NumPy

Polars

openpyxl

Jupyter

PySpark

Domaine : Visualisation de données

Rendre les données visibles, claires et convaincantes.



Graphiques statistiques

Histogrammes, nuages de points et courbes avec Matplotlib et Seaborn.



Tableaux de bord

Plotly Dash et Streamlit : applications interactives en Python pur.



Cartographie

Cartes interactives avec Folium : ventes, trafic, épidémies.

Visualisation de données : exemple en Python



```
import matplotlib.pyplot as plt

mois = ["Jan", "Fév", "Mar", "Avr"]
ventes = [120, 150, 90, 180]
plt.bar(mois, ventes, color="#3776AB")
plt.title("Ventes mensuelles")
plt.show()
```

Bibliothèques clés

Matplotlib

Seaborn

Plotly

Streamlit

Dash

Folium

Un histogramme prêt à présenter

Domaine : Machine learning

Apprendre à partir des données pour prédire et décider.



Prédiction

Prix, ventes, consommation,
abandon de clients.



Classification

Détection de spam, de fraude
bancaire ou de maladies.



Recommandation

Films, produits ou cours adaptés
au profil de chaque utilisateur.

Machine learning : exemple en Python



```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier

X_tr, X_te, y_tr, y_te = train_test_split(X, y)
model = RandomForestClassifier().fit(X_tr, y_tr)
print(model.score(X_te, y_te))
```

Bibliothèques clés

scikit-learn

XGBoost

LightGBM

pandas

MLflow

SHAP

Entraîner et évaluer un modèle

Domaine : Deep learning et vision

Des réseaux de neurones qui voient, écoutent et apprennent.



Vision par ordinateur

Reconnaissance de visages,
d'objets et de plaques
d'immatriculation.



Imagerie médicale

Aide à la détection de
anomalies sur radios et
scanners.



Véhicules autonomes

Perception de la route, des
panneaux et des piétons.

Deep learning et vision : exemple en Python



```
import torch.nn as nn

model = nn.Sequential(
    nn.Linear(784, 128),
    nn.ReLU(),
    nn.Linear(128, 10),
)
print(model)
```

Bibliothèques clés

PyTorch

TensorFlow

Keras

OpenCV

YOLO

Pillow

Un réseau de neurones en 5 lignes

Domaine : NLP et IA générative

Faire comprendre et produire du langage aux machines.



Langues et arabe

Traitement automatique de l'arabe et des langues peu dotées.



Chatbots et assistants

Agents conversationnels basés sur des grands modèles de langage.



Classification de textes

Avis clients, articles de presse, détection de sentiments.

NLP et IA générative : exemple en Python



```
from transformers import pipeline

clf = pipeline("sentiment-analysis")
print(clf("I love learning Python!"))

# [{"label": "POSITIVE", "score": 0.99}]
```

Bibliothèques clés

Transformers

spaCy

NLTK

Gensim

LangChain

scikit-learn

Analyse de sentiment en 3 lignes

Domaine : Automatisation et scripts

Déléguer à Python toutes les tâches répétitives.



Gestion de fichiers

Renommer, trier et sauvegarder des milliers de fichiers.



Excel et e-mails

Générer des rapports et envoyer des messages automatiquement.



Web scraping

Extraire des informations de pages web publiques.

Automatisation et scripts : exemple en Python



```
import shutil
from pathlib import Path

for f in Path("Telechargements").glob("*.pdf"):
    shutil.move(f, "PDF/")
    print("Déplacé :", f.name)
```

Bibliothèques clés

pathlib

openpyxl

BeautifulSoup

Selenium

Requests

schedule

Ranger tous les PDF en un clic

Domaine : Systèmes, DevOps et réseaux

Administrer, superviser et déployer à grande échelle.



Administration système

Scripts de supervision, sauvegardes et gestion des utilisateurs.



DevOps et cloud

Ansible, Docker et boto3 pour automatiser le déploiement.



Réseaux

Configurer des équipements et analyser le trafic avec Netmiko, Scapy.

Systemes, DevOps et reseaux : exemple en Python



```
import psutil

print("CPU :", psutil.cpu_percent(), "%")
print("RAM :", psutil.virtual_memory().percent, "%")

for d in psutil.disk_partitions():
    print("Disque :", d.mountpoint)
```

Bibliothèques clés

psutil

Paramiko

Ansible

Fabric

boto3

Netmiko

Superviser un serveur en quelques lignes

Domaine : Cybersécurité

Détecter, analyser et protéger, dans un cadre éthique et légal.



Analyse de logs

Repérer automatiquement les comportements suspects.



Audit et pentest

Scripts d'audit, scan de ports et tests d'intrusion autorisés.



Cryptographie

Hachage, chiffrement et gestion sécurisée des mots de passe.

Cybersécurité : exemple en Python



```
import hashlib

mdp = "MotDePasse!2025"
empreinte = hashlib.sha256(mdp.encode())
print(empreinte.hexdigest())

# on ne stocke jamais un mot de passe en clair
```

Bibliothèques clés

cryptography

Scapy

python-nmap

Requests

pwntools

YARA

Calculer une empreinte SHA-256

Domaine : Applications et jeux

Créer des interfaces graphiques et des jeux 2D.



Jeux 2D

Pygame : sprites, collisions, sons et scores.



Applications desktop

Tkinter et PyQt pour des logiciels avec fenêtres et boutons.



Outils internes

Petits logiciels sur mesure pour une entreprise ou une école.

Applications et jeux : exemple en Python



```
import pygame

pygame.init()
ecran = pygame.display.set_mode((640, 360))
ecran.fill((255, 212, 59))
pygame.display.flip()
pygame.time.wait(3000)
```

Bibliothèques clés

Pygame

Tkinter

PyQt

Kivy

Arcade

customtkinter

Ouvrir une fenêtre de jeu

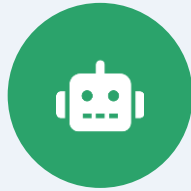
Domaine : IoT et robotique

Relier le monde physique au code : capteurs, moteurs, cartes.



Objets connectés

Capteurs de température, domotique et alertes à distance.



Robotique

Contrôle de robots et simulation avec ROS.



Agriculture intelligente

Irrigation pilotée par l'humidité du sol.

IoT et robotique : exemple en Python



```
from machine import Pin
import time

led = Pin(2, Pin.OUT)
while True:
    led.toggle()
    time.sleep(1)
```

MicroPython : faire clignoter une LED

Bibliothèques clés

MicroPython

gpiozero

ROS

OpenCV

paho-mqtt

pyserial

Domaine : Finance et sciences

Modéliser, simuler et calculer avec précision.



Finance quantitative

Analyse de risque, backtesting
et trading algorithmique.



Recherche scientifique

Physique, biologie, astronomie
avec SciPy, Biopython, Astropy.



Simulation

Modéliser des phénomènes :
population, climat, trafic.

Finance et sciences : exemple en Python



```
import numpy as np

capital, taux, n = 10000, 0.04, 10
valeur = capital * (1 + taux) ** np.arange(n + 1)
print(valeur.round(0))
```

Bibliothèques clés

NumPy

SciPy

SymPy

statsmodels

QuantLib

Biopython

Évolution d'un placement sur 10 ans

03

Filière Développement informatique

Concevoir, programmer et déployer des applications

Ce que vous allez construire

Python est un outil de création



Applications web

Sites dynamiques.



API et services

Back-end de mobiles.



Logiciels

Applications desktop.



Outils

Automatisation métier.

Compétences à développer

1 Algorithmique et structures de données

2 Programmation orientée objet

3 Bases de données

4 Git, tests et bonnes pratiques

5 Conception d'API REST

Exemple : programmation orientée objet



```
class Etudiant:
    def __init__(self, nom, notes):
        self.nom = nom
        self.notes = notes

    def moyenne(self):
        return sum(self.notes) / len(self.notes)

print(Etudiant("Sara", [14, 16]).moyenne())
```

Notions illustrées

Classes et objets

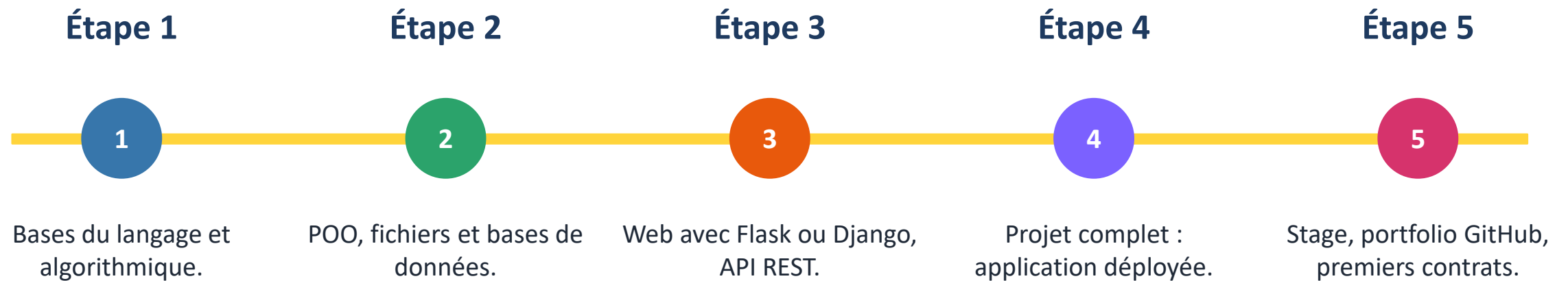
Attributs et méthodes

Encapsulation

Réutilisation du code

Une classe, un objet, une méthode

Parcours type d'un développeur



Métiers visés



Développeur back-end

Logique métier, API, bases de données.



Développeur full-stack

Du serveur à l'interface utilisateur.



Ingénieur DevOps

Déploiement et automatisation.



Testeur / QA

Tests automatisés avec pytest.

Projets pour votre portfolio



Gestion de bibliothèque

CRUD, base de données et interface.



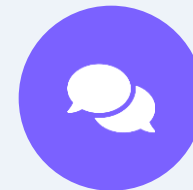
Blog avec Django

Comptes, articles et commentaires.



API de réservation

Services REST documentés.



Bot Telegram

Un assistant utile pour votre classe.

04

Filière Data

Collecter, analyser et valoriser les données

Le cycle d'un projet data

Python accompagne chaque étape

1

Collecter

Fichiers, bases, API, web scraping.

2

Nettoyer

Valeurs manquantes, doublons, formats.

3

Analyser et modéliser

Statistiques et machine learning.

4

Communiquer

Graphiques, rapports, tableaux de bord.

Compétences à développer

1

Statistiques et probabilités

2

Manipulation de données : pandas et SQL

3

Visualisation et storytelling

4

Machine learning

5

Big data et cloud : Spark, Hadoop

Exemple : du fichier CSV à la prédiction



```
df = pd.read_csv("etudiants.csv")
X = df[["heures_etude", "absences"]]
y = df["reussite"]

model = LogisticRegression().fit(X, y)
print(model.predict([[12, 2]]))
```

Outils utilisés

pandas

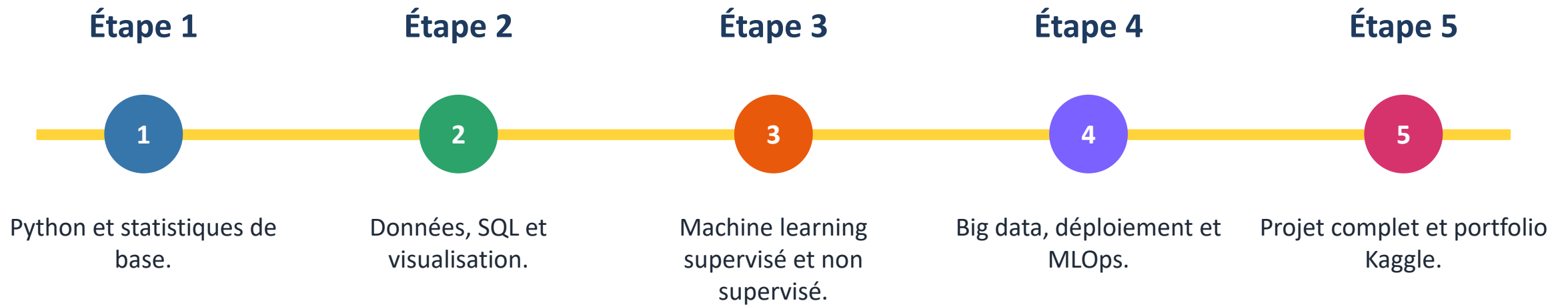
scikit-learn

Jupyter

Matplotlib

Prédire la réussite d'un étudiant

Parcours type d'un profil data



Métiers visés



Data analyst

Indicateurs et tableaux de bord.



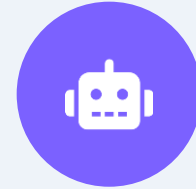
Data scientist

Modèles prédictifs.



Data engineer

Pipelines et entrepôts de données.



Ingénieur IA / ML

Modèles en production.

Projets pour votre portfolio



Tableau de bord des ventes

Streamlit ou Dash.



Prédire la réussite

Classification sur données réelles.



Avis clients

Analyse de sentiments (NLP).



Concours Kaggle

Se mesurer à la communauté.

05

Démarrer avec Python

Installer, pratiquer, progresser

Installer son environnement

Quatre étapes, moins de 30 minutes

1

Installer Python 3

Téléchargement gratuit
sur python.org.

2

Choisir un éditeur

VS Code ou PyCharm
Community.

3

Jupyter ou Colab

Pour tester sans rien
installer.

4

Environnement virtuel

`python -m venv` pour
isoler chaque projet.

Votre premier programme



```
# premier.py
prenom = "Amina"

for i in range(3):
    print(f"{i + 1}. Bonjour {prenom} !")
```

Comment l'exécuter

python premier.py

Ctrl+Entrée (Jupyter)

Bouton  (VS Code)

Sortie : trois lignes de salutations

Les 4 notions à maîtriser d'abord



Variables et types

```
age = 19 ; nom = "Sara"
```



Conditions

```
if note >= 10:  
    print("Admis")
```



Boucles

```
for i in range(5): ...
```



Fonctions

```
def carre(x): return x * x
```

Bien apprendre : la méthode

1 Pratiquer chaque jour : 30 min valent mieux que 5 h

2 Faire de petits projets, dès le début

3 Lire le code des autres

4 Utiliser Git et GitHub dès la première semaine

5 Accepter les erreurs : on apprend en déboguant



Ressources gratuites



Documentation officielle

docs.python.org et son tutoriel.



Cours en ligne

OpenClassrooms,
Coursera,
freeCodeCamp.



Exercices

HackerRank, Codewars,
Kaggle Learn.



Communautés

Stack Overflow, forums
et groupes d'étudiants.

Les pièges à éviter



Copier sans comprendre

Réécrivez chaque solution avec vos mots.



Apprendre sans pratiquer

La lecture seule ne suffit pas.



Sauter les bases

Elles servent dans tous les domaines.



Rester seul

Posez des questions, rejoignez un groupe.

06

Pourquoi vous lancer maintenant ?

Un domaine riche, en pleine expansion

Un marché qui recrute



Transformation numérique

Toutes les entreprises ont besoin de données et d'automatisation.



IA omniprésente

Et Python en est le langage principal.



Compétence rare

Les profils qui maîtrisent Python sont très recherchés.



Télétravail et freelance

Un métier qui s'exerce partout.

Ce que Python vous apporte



Un langage pour la vie

Il reste utile quel que soit votre futur métier.



Une façon de penser

Décomposer un problème, structurer une solution.



Un tremplin vers l'IA

La porte d'entrée vers le machine learning et les LLM.

Pas besoin d'être un génie

Il suffit de commencer, de pratiquer régulièrement et de persévérer.

Mythes et réalités



« Il faut être fort en maths »

Réalité : la logique suffit pour démarrer. Les maths viennent avec la pratique.



« Python est trop lent »

Réalité : les bibliothèques clés sont écrites en C et très rapides.



« Il est trop tard »

Réalité : tout le monde a commencé un jour, et le domaine évolue sans cesse.

Merci !

Place à vos questions