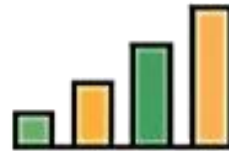




Python pour l'Aménagement et l'Urbanisme



Folium



matplotlib



GeoPandas

Plan du cours

| Séance | Contenu | Objectifs |
|----------|---|--|
| Séance 1 | Introduction à la programmation & Python - Installation et configuration de l'environnement | Comprendre la notion de langage de programmation et mettre en place l'environnement Python nécessaire pour développer des projets en urbanisme. |
| Séance 2 | Variables, types de données et opérations - Manipulations de nombres et de textes | Maîtriser les bases du calcul et du traitement de chaînes de caractères pour préparer des données d'aménagement. |
| Séance 3 | Conditions et structures de contrôle - Utilisation des instructions conditionnelles (if, elif, else) | Implémenter la logique conditionnelle dans des programmes afin de modéliser des scénarios d'analyse urbaine. |
| Séance 4 | Boucles for et while - Utilisation des boucles pour automatiser les répétitions | Automatiser des opérations répétitives pour le traitement de données urbaines en masse. |
| Séance 5 | Fonctions : définition et utilisation - Création et appel de fonctions réutilisables | Structurer et modulariser le code en créant des fonctions pour favoriser la réutilisation et la maintenance. |
| Séance 6 | Listes, tuples et dictionnaires - Gestion d'ensembles de données urbaines simples | Manipuler et organiser efficacement des collections de données issues du domaine urbain (par exemple, statistiques démographiques ou inventaires d'équipements). |

| Séance | Contenu | Objectifs |
|-----------|---|---|
| Séance 7 | Introduction à NumPy - Opérations sur des tableaux de données numériques | Utiliser NumPy pour traiter rapidement de grands ensembles de données numériques liés à l'aménagement. |
| Séance 8 | Introduction aux fichiers CSV et Pandas - Lecture et écriture de fichiers - Traitement de jeux de données | Importer, explorer et transformer des données urbaines (ex. : données de population, équipements publics) grâce à Pandas. Étude de cas : Analyse des équipements publics d'une commune. |
| Séance 9 | Visualisation de données avec Matplotlib - Création de graphiques pour illustrer des analyses | Réaliser des visualisations (graphiques, histogrammes, courbes) afin de présenter des résultats d'analyse de manière claire et impactante. |
| Séance 10 | Analyse spatiale avec GeoPandas - Gestion de données géospatiales (formes, coordonnées) | Exploiter GeoPandas pour traiter et analyser des données géographiques de régions et quartiers, en vue d'études d'aménagement. |
| Séance 11 | Cartographie interactive avec Folium - Création de cartes interactives pour visualiser des projets urbains | Développer des cartes interactives pour présenter des analyses de quartiers et de zones urbaines de manière dynamique et intuitive. |
| Séance 12 | Mini-projet ou QCM final - Projet pratique : Analyse d'un quartier marocain à partir de données réelles | Mettre en application l'ensemble des compétences acquises en réalisant une étude de cas sur un quartier, incluant la collecte, l'analyse et la visualisation des données, afin de présenter un projet d'aménagement urbain. |



Formulaire

<https://forms.gle/BjjTy1pfq85tDW7V6>





Séance 1 : Introduction à la programmation & Python

Objectifs :

- ✓ Comprendre ce qu'est un langage de programmation
- ✓ Découvrir Python et son usage dans l'aménagement et l'urbanisme
- ✓ Installer Python et un environnement de travail
- ✓ Écrire et exécuter son premier programme Python

Qu'est-ce qu'un langage de programmation ?



Qu'est-ce qu'un langage de programmation ?

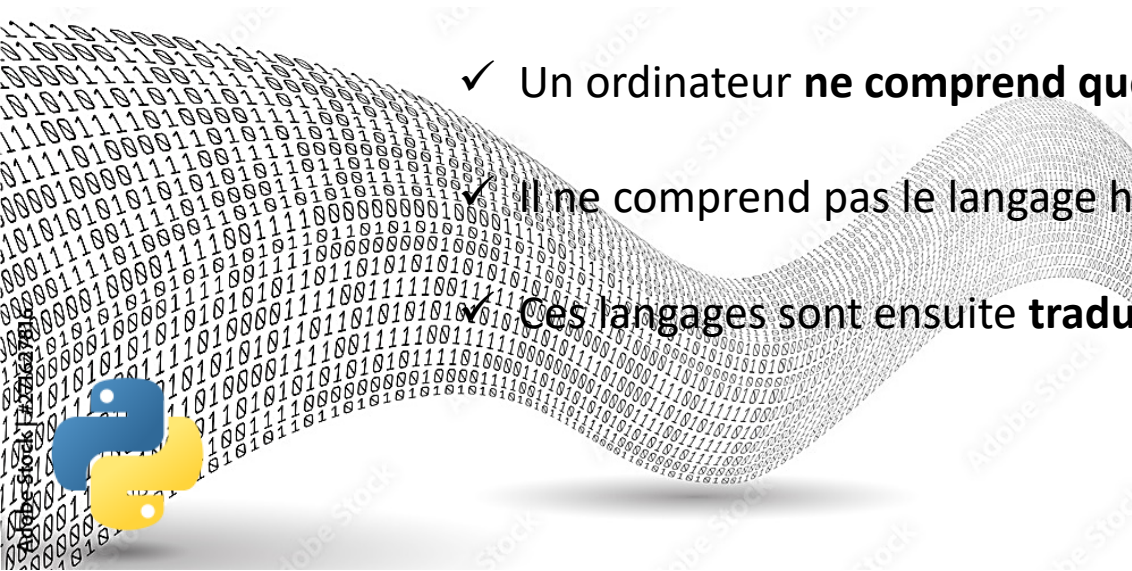
Un **langage de programmation** est un langage utilisé pour donner des instructions à un **ordinateur** afin qu'il effectue des tâches spécifiques.

C'est un **moyen de communication** entre l'humain (le programmeur) et la machine (l'ordinateur).

```
1 requests.get(url)
2 # checking response.status_code (if you get 502, try rerunning the code)
3 if response.status_code != 200:
4     print(f"Status: {response.status_code} - Try rerunning the code!")
5 else:
6     print(f"Status: {response.status_code}\n")
7
8 # using BeautifulSoup to parse the response object
9 soup = BeautifulSoup(response.content, "html.parser")
10
11 # finding Post images in the soup
12 images = soup.find_all("img", attrs={"alt": "Post image"})
13
14 # downloading images
15 for i in range(len(images)):
16     # downloading images
```

Pourquoi utiliser un langage de programmation ?

- ✓ Un ordinateur **ne comprend que des instructions précises**.
- ✓ Il ne comprend pas le langage humain, donc on utilise des **langages structurés**, avec des règles.
- ✓ Ces langages sont ensuite **traduits** (compilés ou interprétés) en **langage machine** (des 0 et des 1).

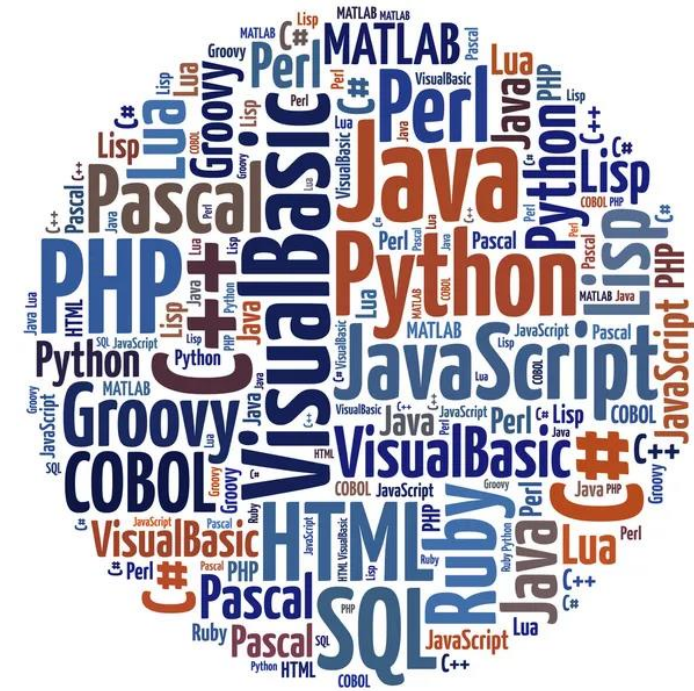


Langages populaires :

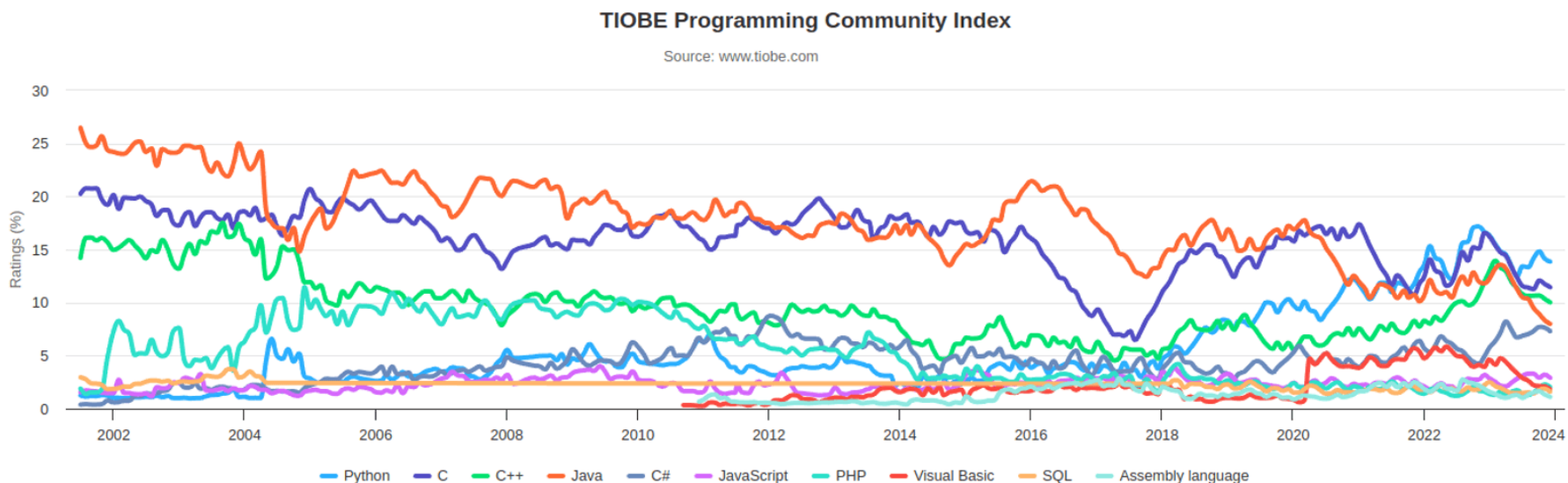
- ✓ **Python** : simple et facile à apprendre (idéal pour les débutants).
- ✓ **Java** : souvent utilisé pour les applications d'entreprise.
- ✓ **JavaScript** : pour les sites web interactifs.
- ✓ **C / C++** : rapides, utilisés dans les systèmes et logiciels embarqués.
- ✓ **R** : pour les statistiques et l'analyse de données.

Chaque langage a ses spécificités :

- ✓ Certains sont **plus faciles à apprendre**.
- ✓ D'autres sont faits pour des usages **plus techniques** ou **plus proches du matériel**.



Top 10 des langages de programmation de 2025



| Jan 2025 | Jan 2024 | Change | Programming Language | | Ratings | Change |
|----------|----------|--------|---|--------------|---------|--------|
| 1 | 1 | |  | Python | 23.28% | +9.32% |
| 2 | 3 | ▲ |  | C++ | 10.29% | +0.33% |
| 3 | 4 | ▲ |  | Java | 10.15% | +2.28% |
| 4 | 2 | ▼ |  | C | 8.86% | -2.59% |
| 5 | 5 | |  | C# | 4.45% | -2.71% |
| 6 | 6 | |  | JavaScript | 4.20% | +1.43% |
| 7 | 11 | ▲ |  | Go | 2.61% | +1.24% |
| 8 | 9 | ▲ |  | SQL | 2.41% | +0.95% |
| 9 | 8 | ▼ |  | Visual Basic | 2.37% | +0.77% |
| 10 | 12 | ▲ |  | Fortran | 2.04% | +0.94% |



Python

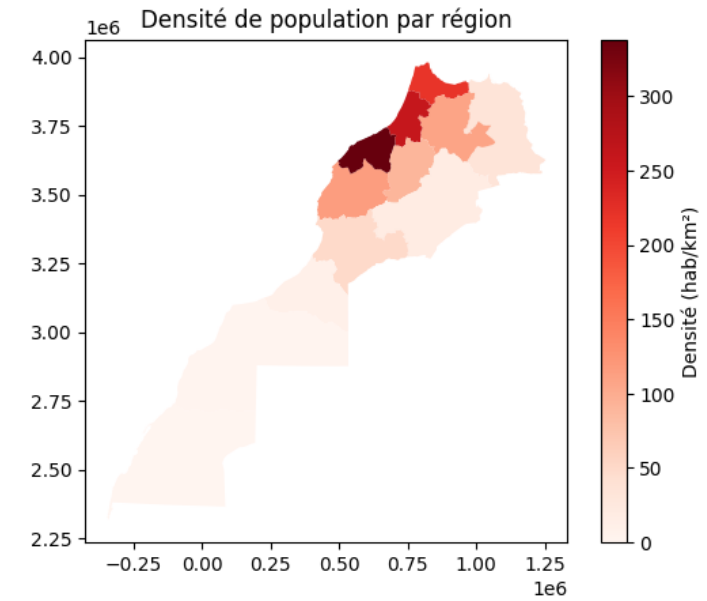
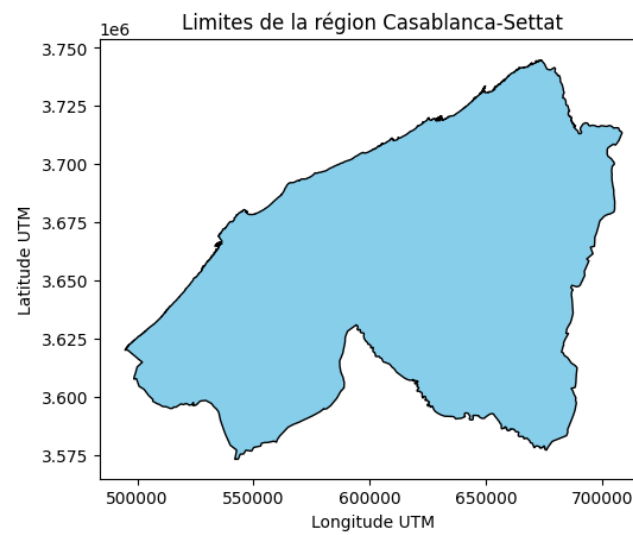
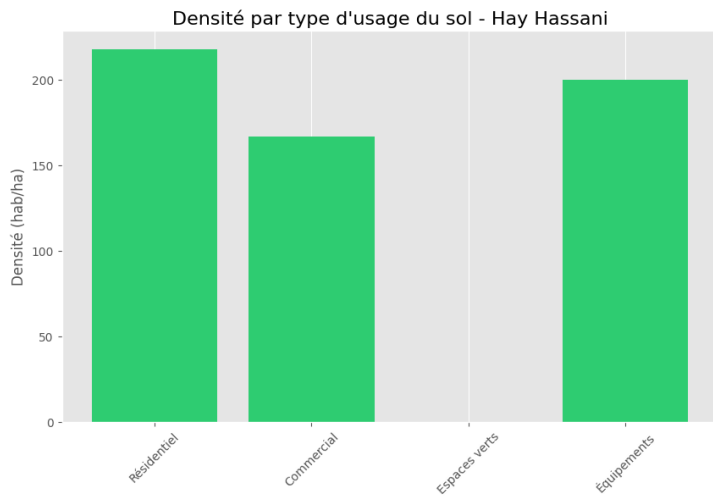
Qu'est-ce que Python

- ✓ Python est un **langage de programmation** très populaire, simple à lire et à écrire.
- ✓ Créé en 1991, il est utilisé dans **de nombreux domaines** : web, intelligence artificielle, traitement de données, automatisation, géomatique, urbanisme, etc.

Pourquoi Python est-il populaire ?

| Avantages | Détails |
|--------------------|---|
| Simple | Syntaxe claire, facile à lire |
| Polyvalent | Utilisé dans la science des données, le web, la cartographie... |
| Rapide à apprendre | Parfait pour les débutants |
| Communauté active | Beaucoup de ressources, d'outils, d'aides en ligne |





Pourquoi Python pour l'aménagement et l'urbanisme ?

- ✓ Analyse de données urbaines (densité, mobilité, accessibilité...)
- ✓ Traitement de données géographiques avec des bibliothèques comme geopandas, shapely, folium, QGIS + Python
- ✓ Modélisation urbaine (ex. : analyser la croissance d'un quartier)
- ✓ Automatisation de tâches répétitives (extraction de données, génération de rapports, etc.)



Installation de Python

Installation simple - Étape par Étape

Télécharger Python

Aller sur le site officiel : <https://www.python.org/downloads/>

- ✓ Le site détecte automatiquement votre système (Windows, macOS, Linux).
- ✓ Cliquez sur "**Download Python 3.x.x**" (ex : 3.13.3)

Pour les utilisateurs Windows

- ✓ Ouvrir le fichier téléchargé python-3.x.x.exe
- ✓ Cochez la case "Add Python to PATH"
- ✓ Cliquez sur "Install Now"
- ✓ Une fois terminé, cliquez sur "Close"

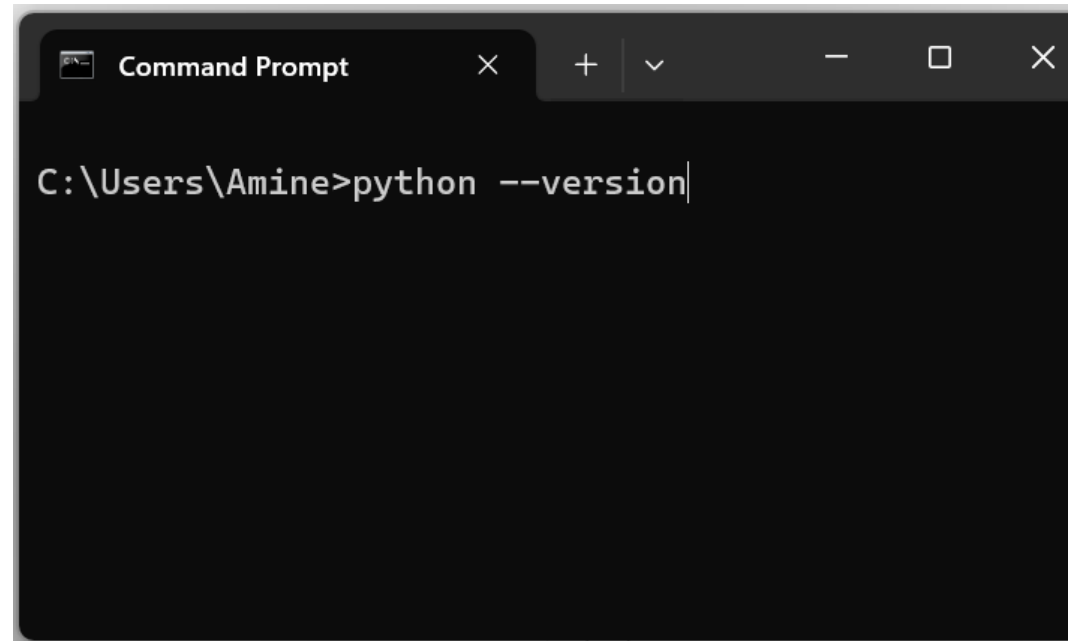


Installation de Python

Installation simple - Étape par Étape

Tester l'installation :

- ✓ Ouvrir le menu Démarrer → chercher **cmd**
- ✓ Écrire dans la console :



```
Command Prompt
C:\Users\Amine>python --version
```



Installation de Python

Environnement de Développement Intégré (IDE)

Un **Environnement de Développement Intégré** (en anglais *Integrated Development Environment* ou **IDE**) est un **logiciel** ou un **ensemble d'outils** regroupés dans une interface unifiée, qui permet aux développeurs de **concevoir, écrire, tester, déboguer et exécuter** des programmes informatiques de manière efficace



Pycharm



VS Code



Thonny



Jupyter Notebook



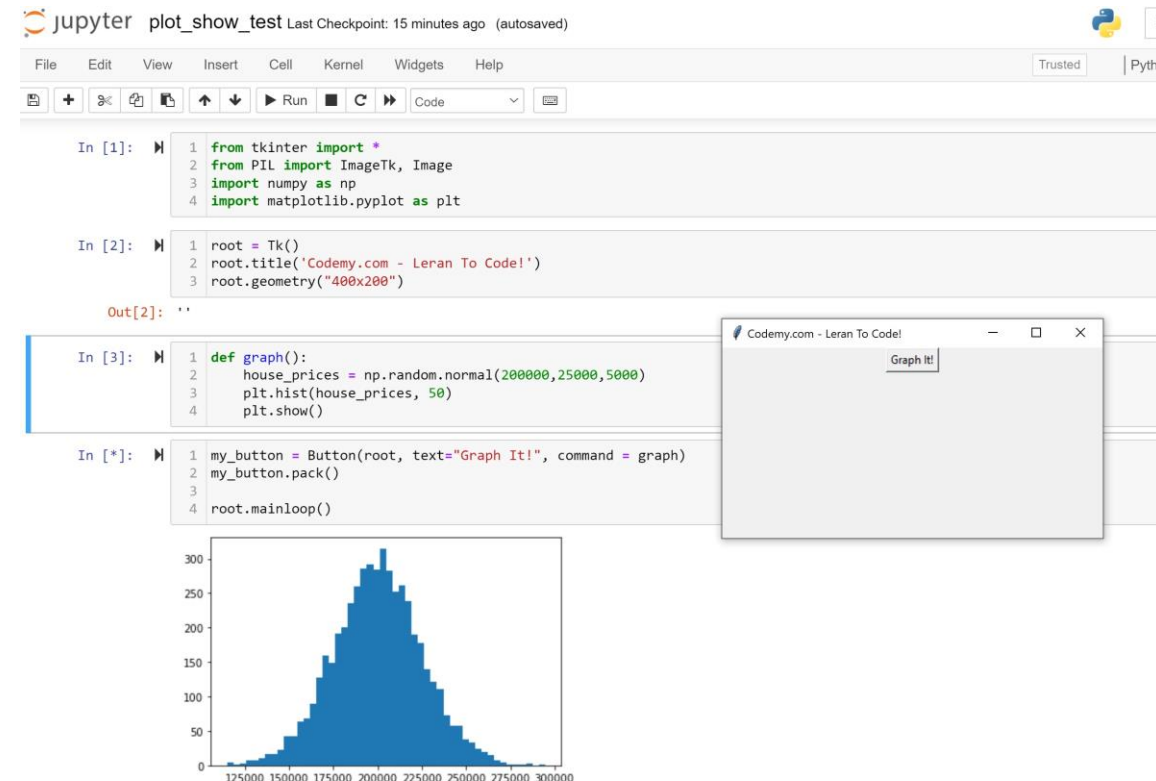
Installation de Python

Jupyter Notebook



Pourquoi l'utiliser ?

- ✓ **Interface claire** : on écrit du texte et du code dans le même document.
- ✓ Permet d'ajouter des **explications en Markdown**, des **graphiques**, des **résultats en temps réel**.
- ✓ Idéal pour tester, expérimenter, faire des démonstrations ou documenter un projet.



Installation de Python

Installation avancée avec Anaconda



Anaconda : Une distribution Python clé en main

- ✓ **Distribution Python complète :**

Anaconda installe **Python** et une vaste **bibliothèque de packages** prédéfinis. (voir la liste de packages Anaconda)

- ✓ **Gestion simplifiée des packages avec Conda :**

L'outil **Conda** permet de **mettre à jour** et **installer facilement** les librairies dont vous avez besoin pour vos développements.



Installation de Python

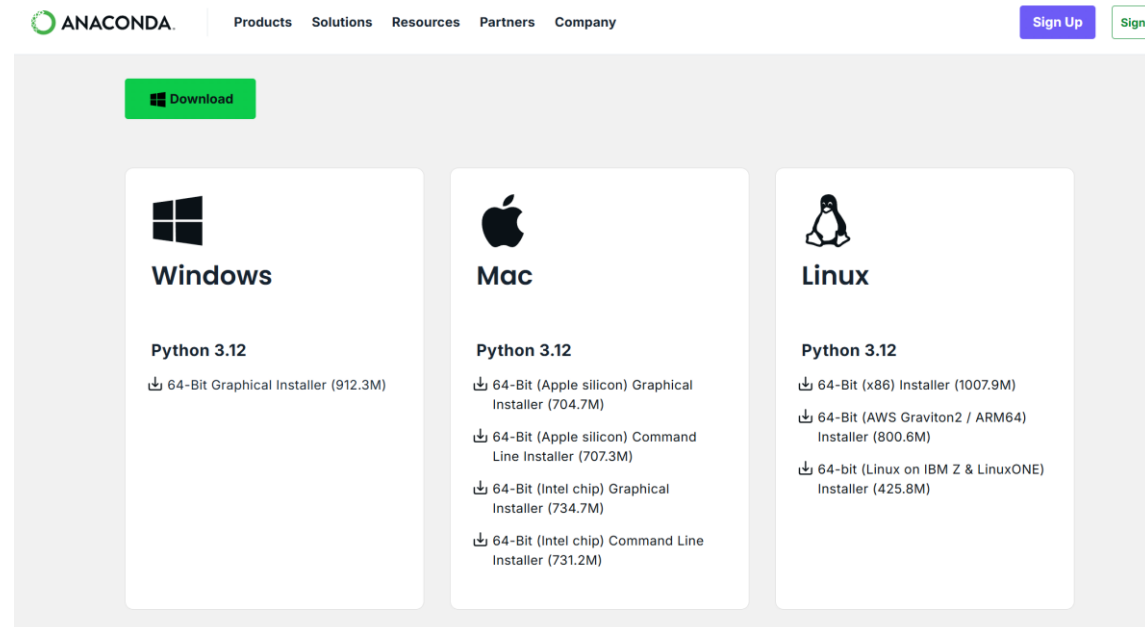
Installation avancée avec Anaconda

Télécharger Anaconda

Rendez-vous sur le site officiel d'Anaconda : <https://www.anaconda.com/products/distribution>

Téléchargez la version pour votre système d'exploitation (Windows, macOS, ou Linux).

- ✓ Choisissez la version **Graphique** (pour Windows, c'est un fichier .exe) pour une installation facile.
- ✓ Choisissez la version **Python 3.x** (de préférence la plus récente).



Installation de Python

Installation avancée avec Anaconda

Installation sur Windows

1. Lancez l'installateur que vous avez téléchargé (**Anaconda3-xxx-Windows-x86_64.exe**).
2. Suivez les étapes d'installation : Cochez la case "**Add Anaconda to my PATH environment variable**" (recommandé pour simplifier l'accès).
3. Choisissez l'option "**Install for me only**" pour une installation personnalisée (ou "**All users**" si nécessaire).
4. Cliquez sur "**Install**" et attendez que l'installation se termine.
5. Cliquez sur "**Next**" et "**Finish**" pour terminer.



Installation de Python

Vérification de l'installation

Après l'installation, vérifiez que Anaconda et Python sont bien installés en ouvrant un terminal ou une invite de commande et en tapant :

```
Command Prompt
Microsoft Windows [Version 10.0.22631.5126]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amine>conda --version
```

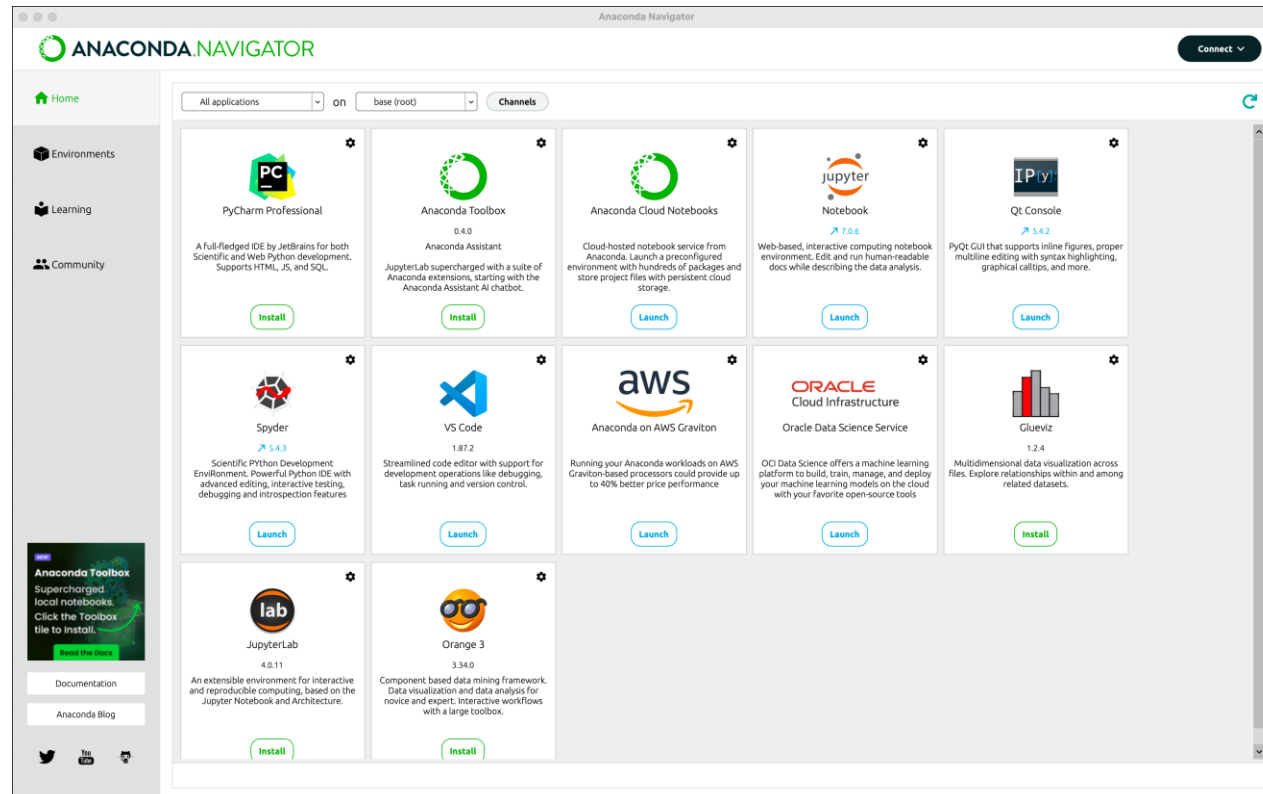


Installation de Python

Utilisation de Anaconda Navigator

Anaconda inclut **Anaconda Navigator**, une interface graphique qui permet de gérer facilement vos environnements Python et d'installer des bibliothèques.

1. Ouvrez **Anaconda Navigator** à partir du menu Démarrer (Windows) ou via la recherche macOS.
2. Vous pouvez maintenant utiliser **Jupyter Notebook**, **PyCharm**, **Spyder**, ou **VS Code** pour coder en Python.



ANACONDA®

Installation de Python

Créer un environnement virtuel avec Anaconda

Il est recommandé de travailler avec des **environnements virtuels** pour éviter les conflits de dépendances entre les projets.

- ✓ Ouvrez un terminal et tapez la commande suivante pour créer un nouvel environnement Python :

```
Command Prompt
Microsoft Windows [Version 10.0.22631.5126]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amine>conda create --name monenv python=3.13|
```



Installation de Python

Créer un environnement virtuel avec Anaconda

Pour activer l'environnement :

```
Command Prompt
Microsoft Windows [Version 10.0.22631.5126]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amine>conda activate monenv
```



Installation de Python

Installer des packages avec Anaconda

Anaconda vous permet d'installer facilement des packages et des bibliothèques avec **conda** :

```
Command Prompt
Microsoft Windows [Version 10.0.22631.5126]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amine>conda install numpy matplotlib pandas geopandas folium
```



Test final

Pour tester l'installation d'Anaconda et Python, lancez **Jupyter Notebook** via Anaconda Navigator et créez un nouveau notebook Python. Tapez : **print("Bonjour INAU")**

