Formation Python: la plus complète

Introduction à Python

- Installer Python et un éditeur (VS Code, Jupyter…)
- Découvrir la console interactive et les scripts
- Créer son premier script Python

Syntaxe et fondamentaux

- Types de données (int, float, str, bool)
- Variables et typage dynamique
- Opérations arithmétiques et logiques
- Conditions (if, elif, else)
- Boucles (for, while) et itérations
- Fonctions (définition, paramètres, retour)
- Introduction au typage statique avec type hints

Manipulation de chaînes de caractères et de fichiers

- Opérations sur les chaînes de caractères
- Lecture et écriture de fichiers avec with open()
- Manipulation de fichiers CSV
- Introduction au format JSON

Structures de données

- Listes, tuples, ensembles, dictionnaires
- Parcours et filtrage avec list comprehension
- Fonctions lambda, map, filter, zip
- Tri et recherche dans les structures

Gestion des erreurs et exceptions

- Comprendre les types d’erreurs (SyntaxError, ValueError…)
- Utiliser try / except / finally
- Lever des exceptions personnalisées
- Bonnes pratiques de gestion d’erreurs

Programmation modulaire

- Créer et utiliser des modules
- Importer avec import et from
- Découverte des modules standards utiles (os, datetime, pathlib, etc.)

Programmation orientée objet (POO)

- Créer une classe, instancier des objets

- Encapsulation, héritage, polymorphisme
- Constructeurs et méthodes spéciales (__init__, __str__…)
- Introduction aux dataclasses

Interaction avec des bases de données

- Connexion à SQLite avec sqlite3
- Lire, insérer, mettre à jour et supprimer des données
- Introduction à SQLAlchemy (ORM moderne)

Travailler avec des API Web

- Requêtes HTTP avec requests
- Consommer une API REST (GET, POST…)
- Analyse des réponses JSON

Développement Web avec Flask

- Principes de base de Flask
- Création d'un serveur REST simple
- Routage et gestion des requêtes
- Tester une API avec Postman ou curl

Tests automatisés avec Pytest

- Écrire des tests unitaires avec pytest
- Asserts, organisation des fichiers de test
- Exécuter les tests et interpréter les résultats

Initiation à la manipulation de données

- Introduction à NumPy et Pandas
- Lire des fichiers CSV avec Pandas
- Filtrer, grouper et résumer des données
- Création de DataFrames à partir de données JSON ou SQL

Démarrer avec Python : des fondamentaux à l'orienté objet

Introduction à la formation Python

- Présentation de Python et de son écosystème
- Origine, philosophie et cas d'usage
- Installation de Python et prise en main de l'IDE

- Découverte de la documentation officielle et des ressources en ligne
- Introduction aux modes d'exécution : script et interactif
- Premier programme : « Hello World »

Module 1 : Syntaxe de Python

- Introduction à la formation Python
- Types et affectation : int, float, str, bool
- Opérateurs arithmétiques et logiques
- Calculs et opérations sur les variables
- Les commentaires en Python
- Les entrées/sorties (input, print, formatage)
- Instructions de contrôle : pass, if, elif, else, while, for
- La fonction range et les itérations contrôlées
- Chaînes de caractères et slicing
- Fonctions prédéfinies et création de fonctions
- Les fichiers : ouvrir un fichier, modes d'ouverture
- Lire et écrire dans un fichier texte
- Gestion de l'encodage
- Atelier pratique : *Création de scripts complets*

Module 2 : Gestion des erreurs / Exception

- Comprendre les types d'erreurs : SyntaxError, TypeError, etc.
- Lever et déclencher des exceptions personnalisées
- Gestion des erreurs avec try/except
- Utilisation des clauses else et finally
- Affichage des messages d'erreurs détaillés
- Atelier pratique : *Simulation et gestion des erreurs*

Module 3 : Maîtriser les structures de données

- Créer et manipuler des listes
- Utiliser les dictionnaires
- Découvrir les tuples et les ensembles
- Utiliser les slices sur les séquences
- Parcourir les structures avec enumerate() et zip()
- La méthode items() sur les dictionnaires
- Tri et filtrage des collections
- Compréhensions de listes et de dictionnaires
- Atelier pratique : *Liste en pile et file*

Module 4 : Modules et Packages

- Présentation des modules standards : os, sys, datetime, math
- Création et organisation d'un module
- Importer des modules : import, from-import
- Découverte des packages et du fichier __init__.py
- Structure d'un projet Python modulaire

- Utiliser pip pour installer des bibliothèques externes
- Atelier pratique : *Créer et tester un module*

Module 5 : Programmation orientée objet en Python

- Concepts fondamentaux : objet, classe, instance
- Créer une première classe avec attributs et méthodes
- Encapsulation et conventions de nommage (_privé, __très_privé)
- Les méthodes spéciales : __init__, __str__, __repr__
- Notion d'héritage et polymorphisme (survol)
- Faire la différence entre composition et agrégation.
- Fonctions à arguments variables (*args, **kwargs)
- Réutilisation du code et bonnes pratiques orientées objet
- Erreurs et exceptions orientées objet
- Atelier pratique : *Création de classes et mise en relation*

Module 6 : Base de données

- Introduction aux bases de données relationnelles
- Installer un driver (sqlite3, MySQL Connector, psycopg2)
- Connexion à une base de données
- Création d'une table avec SQL
- Insérer, supprimer, modifier les données
- Lire et filtrer des données avec des requêtes
- Utiliser des paramètres et prévenir les injections SQL
- Fermer proprement la connexion
- Atelier pratique : *Créer une base et gérer des données*

Formation OpenAI pour développeur Python

Rappel sur les concepts Objets en Python: notions avancées

- Notion d'objet, de classe
- Encapsulation dans Python
- Les décorateurs
- L'essentiel de PEP8
- Erreurs et Exceptions
- Atelier pratique: créer les premières classes Python

Relation entre objets – classes – Python – notions avancées

- Héritage dans Python

- Association
- Atelier pratique: créer des classes et les mettre en relation

Concepts indispensables de Python

- Fonction avec args* et kwargs**
- Expression lambda dans Python
- Fonctions: map, reducer, filter
- Les compréhensions
- Itérateurs, générateurs dans Python

Gestion des erreurs / Exception dans Python

- Connaitre les erreurs
- Lever des exceptions dans Python
- Gestion avec try/except
- Les assertions

Introduction Sciences des données / IA

- Numpy
- Apprentissage automatique

OpenAI pour les développeurs

Comprendre l'Intelligence Artificielle
Comprendre la notion de modèle
Configuration du projet et création d'une clé API

Utilisation des requêtes avec OpenAI

Interaction avec la machine en utilisant l'API ChaGPT
Génération d'images avec l'IA Génératrice et les modèles DALL-E
Donner des instructions à l'IA pour générer des images

Compréhension de ChatGPT

Analyse et explication de code avec ChatGPT
Génération de code avec l'aide de ChatGPT
Introduction au Fine-tuning

Conclusion sur le cours Python et OpenAI pour les développeurs

Les bases de Python

Python c'est quoi ?

Python est un langage de programmation populaire et polyvalent, connu pour sa facilité de lecture, d'écriture et d'apprentissage.

À la différence d'HTML, CSS ou JavaScript, Python n'est pas limité à l'utilisation dans le développement web. Il peut être employé pour différents types de programmation et de développement logiciel

Les avantages de Python :

- **Facilité d'apprentissage :** Python est un langage de programmation facile à apprendre et à comprendre, avec une syntaxe simple qui facilite la lecture et l'écriture du code.
- **Large communauté de développeurs :** Python est l'un des langages de programmation les plus populaires au monde, avec une grande communauté de développeurs actifs qui partagent des connaissances et des ressources en ligne.
- **Bibliothèques et frameworks :** Python dispose d'une grande variété de bibliothèques et de frameworks qui peuvent être utilisés pour développer des applications web, des applications de traitement de données et des projets de machine learning.
- **Portabilité :** Python est un langage de programmation portable, ce qui signifie que le code écrit en Python peut être exécuté sur de nombreuses plateformes, y compris les systèmes d'exploitation Windows, Mac et Linux.
- **Polyvalence :** Python est un langage de programmation polyvalent, qui peut être utilisé pour développer une grande variété d'applications, allant des scripts simples aux applications web complexes en passant par les projets de machine learning.
- **Grande popularité auprès des entreprises et des universités :** Python est largement utilisé par de nombreuses entreprises et institutions, ce qui rend les compétences en Python très précieuses sur le marché du travail.

À quoi sert Python dans le développement Web ?

Python est un langage de programmation polyvalent qui peut appliqué dans de nombreux domaines différents.

- **Applications de machine learning :** Python permet d'implémenter des algorithmes de machine learning pour effectuer des tâches telles que la

reconnaissance faciale, la reconnaissance de l'écriture manuscrite et l'exploration de données.

- **Analyse et visualisation de données :** La partie la plus cruciale de toute entreprise est l'extraction et l'analyse de données. Python offre des bibliothèques et des outils pour extraire les préférences, les goûts et les aversions des utilisateurs. Il y a également des outils statistiques et graphiques qui permettent de visualiser et d'analyser les données pour générer des bénéfices.
- **Prototypage rapide :** Python est idéal pour la création de prototypes d'application. Les programmeurs s'en servent pour présenter les fonctionnalités de base aux clients avant de coder l'application entière.
- **Calcul de données volumineuses :** Python permet d'extraire de grandes quantités de données à partir de différentes pages Web sur Internet.
- **Recherche et développement :** Python est utilisé pour la recherche scientifique et l'informatique grâce à ses nombreuses bibliothèques telles que AstroPY et BioPython.
- **Caractéristiques de l'intelligence artificiel :** L'intelligence artificielle est un domaine qui implique une recherche massive.
- Python à travers ses bibliothèques, ses outils visuels, sa simplicité et sa cohérence représente un avantage considérable par apport aux autres langages.
  La possibilité d'intégrer des Chatbots, la sécurité biométrique, des recommandations personnalisées et même un agent vocal comme Siri ou Alexa fait la différence.
- **Data Science :** La compétence la plus importante pour les data scientists travaillant dans le domaine de la Data Science est Python. La manipulation de vastes ensembles de données est un impératif dans ce domaine. Heureusement, grâce à l'intégration de Python, il est possible de travailler avec des bases de données déjà existantes.
- **Le secteur financier :** Python est beaucoup utilisé dans le secteur financier, non seulement par les sociétés FinTech, mais aussi par d'autres entreprises. Grâce à la flexibilité de Python, il est possible de concevoir des applications d'entreprise qui s'intègrent facilement à des systèmes existants tels que des bases de données, des sites Web ou des applications non Web.
- **Développement de startups :** Python est un choix idéal pour les startups et les petites entreprises en raison de son évolutivité. Les avantages tels que l'évolutivité élevée, les prototypes rapides, le développement rapide de produits minimum viables (MVP) et la rentabilité contribuent à faire de Python un choix privilégié pour ces entreprises.

Les bases de Python

Python est un langage de programmation à la syntaxe simple, ce qui le rend facile à lire et à écrire.

Voici quelques-unes des principales caractéristiques de la syntaxe Python:

- Variables: pour déclarer une variable en Python, il vous suffit d'écrire le nom de la variable suivi du signe égal et de la valeur que vous souhaitez lui affecter.

```
prenom = 'Jean'
age = 30
```

- Structures de contrôle: Python utilise des structures de contrôle telles que les boucles for et while, ainsi que les instructions conditionnelles if, elif et else.

```
if age < 18:
    print('Mineur')
elif age >= 18 and age < 60:
    print('Adulte')
else:
    print('Senior')
```

- Fonctions: les fonctions sont définies en utilisant le mot-clé def, suivi du nom de la fonction et de la liste des paramètres entre parenthèses. Les instructions de la fonction doivent être indentées.

```
def dire_bonjour(prenom):
    print(f'Bonjour, {prenom}!')

dire_bonjour('Jean')
```

Environnement de développement

Pour commencer à travailler avec Python, vous devez installer l'interpréteur Python sur votre ordinateur. Vous pouvez télécharger l'interpréteur Python à partir du site officiel: Python.org

En ce qui concerne les environnements de développement, vous pouvez choisir un éditeur de texte simple comme Sublime Text ou Visual Studio Code, ou un IDE dédié comme PyCharm.

Ressources pour apprendre Python

Des formations proposées par Doussou Formation offrent des cours adaptés à différents niveaux.

Nous formations Python

En clair, Python est un langage de programmation puissant et polyvalent, idéal pour les débutants et les développeurs chevronnés. Avec sa syntaxe simple, sa grande communauté d'utilisateurs et sa polyvalence, Python est le choix parfait pour ceux qui cherchent à se lancer dans le monde de la programmation ou à approfondir leurs compétences. Il ne fait aucun doute que l'apprentissage de Python sera un atout précieux pour votre carrière et vos projets personnels.

Formation Python et science des données – Tour complet

Introduction à la formation Python et science des données – Bloc 1

- Les types de variables (entier, décimal, booléen, etc.) et les opérateurs de base;
- Les structures de données (liste, tuple, dictionnaire, etc.);
- Les contrôles de flux (if-else, try-except-finally);
- Les boucles (for, while);
- Les combinaisons contrôle-boucle;
- Les fonctions et méthodes (fonction(objet) et objet.méthode());
- L'importation de modules externes pour accéder à plus de fonctions-méthodes;
- La syntaxe et les bonnes pratiques.

Introduction à Python – Bloc 2

- Gérer les modules;
- Administrer le système d'exploitation;
- Tirer des donnée du web (web scraping);
- Rectifier les données avec le module regex;
- Intégrer le temps avec le module datetime.

Initation à la science des données

- Jupyter Notebook, Spyder et les autres;
- Introduction à Numpy;
- Introduction à Pandas – Objet Series;
- Introduction à Pandas – Objet DataFrame;
    - Extra: Introduction à Pandas – Instructions;
    - Extra: Tidy Data;
- Groupby et agrégations;
- Cueillette de données avec API Web et analyse;
- Visualisation graphique;
    - Extra: Visualisation cartographique.

Formation Python : finance et séries chronologiques

Introduction à la formation Python : finance et séries chronologiques

Module 1: Introduction à Python pour la Finance

- Présentation de Python en tant qu'outil de programmation pour la finance.
- Installer et configurer l'environnement Python avec les librairies financières.
- Manipuler les données financières, les types de données et les opérations de base.

Module 2: Analyse de données financières avec Pandas

- Introduction à la librairie Pandas pour l'analyse de données.
- Charger des données financières à partir de différentes sources (fichiers CSV, bases de données, etc.).
- Nettoyer et prétraiter les données pour une analyse approfondie.
- Effectuer des opérations de groupement, filtrage, et des calculs

statistiques sur les séries chronologiques.

Module 3: Visualisation de données financières avec Matplotlib et Seaborn

- Introduction à Matplotlib et Seaborn pour la visualisation de données.
- Créer des graphiques tels que les courbes de prix, les histogrammes de rendements, et les graphiques en chandeliers.
- Personnaliser les graphiques pour une présentation visuelle efficace des données financières.

Module 4: Analyse de séries chronologiques financières

- Comprendre les concepts clés des séries chronologiques financières (volatilité, tendances, saisonnalité, etc.).
- Appliquer des modèles de séries chronologiques tels que la moyenne mobile, l'autorégression (AR), la moyenne mobile autorégressive (ARMA) et la moyenne mobile intégrée autorégressive (ARIMA).
- Évaluer et interpréter les performances des modèles.

Module 5: Prédiction des prix financiers avec les modèles de Machine Learning

- Introduction aux concepts de base de l'apprentissage automatique et de la régression.
- Utilisation de bibliothèques de Machine Learning telles que Scikit-Learn pour prédire les prix financiers.
- Comparaison des performances des modèles de régression pour des prédictions précises.

Module 6: Applications pratiques en Finance avec Python

- Appliquer les compétences acquises pour résoudre des problèmes financiers concrets.
- Créer des stratégies d'investissement basées sur l'analyse de séries chronologiques financières.
- Mettre en œuvre des analyses de risque et de portefeuille pour la prise de décisions éclairées.

Formation Django – Framework Python pour le web

Introduction à la formation Django

Rappel rapide sur les concepts avancés en Python

Fonction lambda
Fonctions avec arguments multiples: *arg, **kwargs
Notions d'objets, classes
Notions de packages, importation
Gestion des exceptions.

Introduction au framework Django

Présentation de Django
Présentation de l'architecture MVT
Qu'est ce qu'un projet
Qu'est qu'une application ?

Atelier pratique: créer un projet
Atelier pratique: créer une application

Prise en main de Django

Créer des vues
Créer des routes
Passer des données à la vue
Redirection
Créer une url dans le template
Utiliser les expressions régulières dans le routing
Différence entre url et path

Le langage de gabarit de Django

Variables
Filtres
Commentaires
Les conditions
Les boucles
Créer un layout(héritage de template)
ajouter des fichiers statiques

Modèle

Introduction à la modélisation
Les champs de modèles
Les liaisons entre modèles
Réaliser des requêtes
Les modèles dans les vues

L’administration

Mettre en place l’administration
Manipuler nos propres modèles

Les formulaires

Créer un formulaire
Utiliser le formulaire dans la vue
Traitement des données envoyées

Notions avancées:

Les vues génériques
Filtres et tags personnalisés
Context processor

Formation Python avancée – Perfectionnement

Introduction à la formation Python avancée

L’Orienté Objet en Python: notions avancées

- Objet, classes
- Encapsulation dans Python
- Les méthodes magiques – surcharges dans Python
- Les décorateurs
- L’essentiel de PEP8
- Atelier pratique: créer les premières classes Python

Relation entre objets – classes – Python – notions avancées

- Héritage dans Python
- Héritage multiple dans Python
- Agrégation
- Composition
- Erreurs et Exceptions
- Atelier pratique: créer des classes et les mettre en relation

Concepts utiles de Python

- Fonction à arguments variables

- Formatage
- Créer un décorateur personnalisé
- Expression lambda dans Python
- Fonctions: map, reducer, filter
- Les compréhensions
- Itérateurs, générateurs dans Python

Gestion des erreurs / Exception dans Python

- Connaitre les erreurs
- Lever des exceptions dans Python
- Gestion avec try/except
- Les assertions

Tests unitaires avec Pytest (module Python)

- Découvrir les doctests
- Tester une fonction
- Tester une exception
- Les Fixtures
- Atelier pratique: Multiples exemples

Threading dans Python

- Fonctionnement
- Thread, processus
- L'utilisation dans Python

Introduction Sciences des données / IA

- Numpy
- Apprentissage automatique

Formation Python – Les bases pour la Data Science

Module 1 – Introduction à Python

- Présentation du langage et de ses usages scientifiques
- Installation de Python et de l'éditeur de code (IDLE, VS Code, ou Jupyter)
- Premiers pas avec la console Python et les scripts

Module 2 – Fondements de la programmation

- Variables, types de données et conversions
- Chaînes de caractères, nombres et booléens
- Opérations arithmétiques et logiques

Module 3 – Structures de contrôle

- Conditions (if, elif, else)
- Boucles (for, while)
- Notion d'indentation et bonnes pratiques

Module 4 – Fonctions et scripts

- Créer et appeler une fonction
- Passage de paramètres et retour de valeurs
- Structurer son code en blocs logiques

Module 5 – Introduction à la programmation orientée objet (optionnel)

- Comprendre la notion d'objet en Python
- Créer une classe et instancier des objets simples

Module 6 – Manipulation de fichiers

- Lire et écrire des fichiers texte
- Parcourir et traiter des données ligne par ligne

Module 7 – Mise en pratique

- Création d'un petit programme complet combinant variables, boucles et fonctions
- Corrections et conseils personnalisés

Formation : Atelier pratique en intelligence artificielle moderne

Module 1 – Prise en main des assistants IA

- Utiliser ChatGPT, Gemini et Claude dans un contexte professionnel.
- Comparer les réponses selon les outils utilisés.
- Créer des requêtes simples, contextualisées et orientées résultat.
- Générer du texte, des idées, des plans et des explications.

- Corriger, améliorer et reformuler des contenus avec l'IA.

Module 2 – Atelier de conception d'invites

- Comprendre la structure d'une bonne invite.
- Définir un rôle, un contexte, une tâche et un format de sortie.
- Créer des invites avec contraintes et exemples.
- Utiliser l'enchaînement d'invites pour améliorer les résultats.
- Adapter les invites à différents besoins : rédaction, analyse, code, synthèse.

Module 3 – IA générative multimédia

- Générer des images à partir d'une description textuelle.
- Améliorer une description pour obtenir un meilleur résultat visuel.
- Utiliser l'IA pour retranscrire et traduire un contenu audio.
- Générer une voix ou un contenu audio assisté par IA.
- Classer, résumer et structurer un document avec l'IA.

Module 4 – Programmation assistée par IA

- Générer un exemple d'algorithme simple en Python.
- Faire commenter et expliquer du code par une IA.
- Corriger les erreurs dans un programme généré.
- Exporter et tester le code produit.
- Comprendre les limites du code généré automatiquement.

Module 5 – Atelier Machine Learning avec Python

- Créer un environnement simple de travail avec Python.
- Utiliser NumPy pour manipuler des données.
- Créer un premier modèle supervisé simple.
- Comprendre les étapes : données, entraînement, prédiction, évaluation.
- Explorer Scikit-Learn et PyCaret à travers des exemples pratiques.

Module 6 – Exploitation documentaire avec NotebookLM

- Créer un notebook à partir de documents sources.
- Importer des PDF, pages web, vidéos ou fichiers audio.
- Générer des résumés, FAQ, guides d'étude et notes structurées.
- Produire une synthèse fiable à partir de sources fournies.
- Créer un support d'apprentissage ou un podcast généré par IA.

Module 7 – RAG, ancrage et personnalisation

- Comprendre le principe du RAG dans un contexte pratique.
- Ancrer les réponses de l'IA dans des documents ou sources fiables.
- Utiliser les paramètres pour contrôler la sortie générée.
- Créer un assistant personnalisé ou un Gem.
- Identifier les cas d'usage adaptés au RAG et à la personnalisation.

Module 8 – Introduction pratique aux agents IA

- Comprendre le rôle d’un agent autonome.
- Découvrir Gemini CLI et les outils similaires.
- Automatiser une tâche simple avec l’aide de l’IA.
- Explorer les limites et précautions liées aux agents IA.
- Identifier les usages professionnels possibles.