OSD
OPEN DIGITAL SERVICE

Scannez-moi

WHATSAPP
+213 674 05 38 95

SITE WEB
osdigitalis.com

FICHE FORMATION · OSD

Initiation à la programmation Python

De zéro à une application complète : bases du langage, fichiers, objets et analyse de données

30 heures | 15 séances | débutants | projet fil rouge

Une formation progressive pour apprendre à **programmer en Python sans aucune expérience préalable**, jusqu'à réaliser une application complète et analyser des données réelles.

30 h	15 × 2 h	Débutant	80 %
DURÉE	SÉANCES	NIVEAU	PRATIQUE

Description

Python est aujourd'hui le langage le plus utilisé pour apprendre la programmation, automatiser des tâches, développer des services web et analyser des données. Sa syntaxe claire permet d'obtenir des résultats concrets dès la première séance.

La formation part de zéro : qu'est-ce qu'un programme, comment l'écrire, l'exécuter et le corriger. Chaque notion est immédiatement mise en pratique, puis réutilisée dans les séances suivantes. Un **projet fil rouge** relie tous les apprentissages : une application de gestion de notes, d'abord en quelques lignes, puis modulaire, orientée objet, avec sauvegarde et statistiques.

Public visé

- Étudiants (BTS, licence) en informatique ou en reconversion
- Techniciens et administrateurs souhaitant automatiser leurs tâches
- Toute personne motivée voulant apprendre à programmer

Prérequis

- Aucune connaissance en programmation
- Savoir utiliser un ordinateur (fichiers, dossiers, navigateur)
- Un ordinateur Windows, macOS ou Linux par participant

À l'issue de la formation, vous saurez

☐	écrire, exécuter et déboguer des programmes Python dans VS Code ;
☐	manipuler variables, conditions, boucles, listes et dictionnaires ;
☐	structurer son code en **fonctions** et en **modules** réutilisables ;
☐	lire et écrire des fichiers **texte, CSV et JSON** et gérer les erreurs ;
☐	concevoir des **classes** et des objets (programmation orientée objet) ;
☐	analyser un jeu de données avec **pandas** et produire des graphiques avec **Matplotlib** ;
☐	réaliser et présenter une application complète en binôme.

Méthode pédagogique

- Démonstrations courtes suivies de manipulations immédiates : les participants tapent eux-mêmes chaque exemple.
- Une **fiche d'exercices progressifs** par séance, avec un exemple d'exécution attendu pour chaque exercice.
- Un devoir court entre les séances, corrigé collectivement au début de la suivante.
- Exemples tirés de situations concrètes : notes d'étudiants, ventes, stocks, factures, inscriptions.

Programme — 15 séances

1 **Python KESAKO ?** 2 h

Découverte du langage : ce qu'est un programme, pourquoi Python, où on le trouve (IA, web, data, automatisation). Démonstrations visuelles et premier programme écrit par chacun.

`découverte` `premier programme` `positionnement`

2 **Premiers pas : variables et types** 2 h

Installer l'environnement de travail, afficher du texte, stocker des données dans des variables et dialoguer avec l'utilisateur. Lire un message d'erreur sans paniquer.

`VS Code` `print · input` `int · float · str · bool`

3 **Structures conditionnelles** 2 h

Faire décider un programme : comparaisons, choix simples et multiples, combinaison de conditions. Applications : mentions, tarifs, validation de saisies.

`if · elif · else` `and · or · not` `indentation`

4 **Structures itératives** 2 h

Répéter des instructions : boucles `while` et `for`, compteurs et accumulateurs. Applications : tables de calcul, menus, jeu de devinette.

`while` `for · range()` `break`

5 **Les listes** 2 h

Ranger plusieurs valeurs dans une seule variable : indices, ajout, suppression, tri et parcours. Calculs de moyennes, maximums et recherches.

`indices` `append · sort` `sum · max · min`

6 **Dictionnaires et tuples** 2 h

Associer des clés à des valeurs pour retrouver une information par son nom : répertoires, stocks, compteurs de mots. Les tuples pour regrouper des données fixes.

`dict` `clé → valeur` `.items()` `tuple`

7 **Les fonctions (1)** 2 h

Découper un programme en blocs réutilisables : définir une fonction, lui passer des paramètres, récupérer un résultat. Distinguer afficher et renvoyer.

`def` `paramètres` `return`

8 **Fonctions (2) et modules** 2 h

Valeurs par défaut, portée des variables, utilisation des modules `math` et `random`, et création de son propre module pour organiser le code.

`valeurs par défaut` `import` `module personnel`

9 **Les chaînes de caractères** 2 h

Nettoyer, découper, rechercher et mettre en forme du texte : noms saisis n'importe comment, numéros de téléphone, lignes de données, affichages alignés.

`tranches` `split · join` `f-strings`

10 **Lire et écrire des fichiers** 2 h

Sauvegarder ses données pour les retrouver : fichiers texte, tableaux CSV ouverts dans Excel, et format JSON pour enregistrer l'état d'une application.

`open() · with` `csv` `json`

11 **Gérer les erreurs : exceptions** 2 h

Rendre un programme robuste : intercepter les erreurs de saisie ou de fichier, redemander proprement une valeur, signaler ses propres erreurs.

`try · except` `finally` `raise`

12 **Introduction à la programmation objet** 2 h

Modéliser le monde réel avec des classes : attributs, méthodes et objets qui coopèrent. Étudiants, produits, comptes bancaires, paniers.

`class` `__init__` `méthodes`

13 **Bibliothèques : pandas et Matplotlib** 2 h

Installer des bibliothèques avec `pip`, analyser un tableau de données en quelques lignes (filtres, tris, regroupements) et produire des graphiques et des exports Excel.

`pip` `pandas` `Matplotlib`

14 **Projet fil rouge : GestNotes** 2 h

Réaliser en binôme une application modulaire de gestion des notes : classes, saisies robustes, sauvegarde JSON, export CSV et statistiques, développée par étapes validées.

`binômes` `architecture modulaire` `jalons`

15 **Soutenances et évaluation de synthèse** 2 h

Démonstration et explication du projet devant le formateur, puis évaluation individuelle sur machine couvrant l'ensemble de la formation.

`soutenance` `évaluation /20`

Évaluation et validation

- Contrôle continu sur les exercices et les devoirs.
- **Projet** réalisé en binôme et présenté en soutenance (démonstration + questions individuelles).
- **Évaluation de synthèse** individuelle sur machine, notée sur 20.
- Attestation de formation délivrée par OSD — Open Digital Service.

Modalités

Durée	30 heures, en 15 séances de 2 heures (rythme adaptable)
Format	présentiel en salle informatique, ou intra-entreprise / établissement
Effectif	groupes de 8 à 20 participants
Matériel	un poste par participant avec Python 3 et VS Code (installation guidée en séance 2)
Supports remis	fiches de séance, fiches d'exercices, projet complet
Formateur	Mohammed Slimani — développeur fullstack et architecte DevOps, plus de 20 ans d'expérience

Intéressé(e) par cette formation ?

Programme adaptable à votre public et à votre volume horaire (20 h ou 30 h). Contact : contact@osdigitalis.com · WhatsApp +213 674 05 38 95 · osdigitalis.com

Open Digital Service

Mohammed Slimani — Formateur & consultant

+213 (0)6 74 05 38 95 · contact@osdigitalis.com · msslimani@osdigitalis.com

osdigitalis.com · LinkedIn : in/mohammed-slimani-osdigitalis