Smartfuture
Certifications | Training | Technology solutions

FORMATION & CERTIFICATION • PCAP

Programmeur Python – Niveau Associé PCAP

Maîtrisez les bases solides du langage Python afin de développer des applications simples, structurées et conformes aux bonnes pratiques attendues à l'examen PCAP.

Formateurs certifiés Python & PCAP

5 jours 35 h	**Examen** PCAP
Modalité Présentiel Distanciel	**Niveau** Débutant Intermédiaire

INSCRIPTION / RÉSERVATION

Je m'inscris maintenant

OBJECTIFS PÉDAGOGIQUES

- Comprendre la syntaxe de base du langage Python et son environnement d'exécution.
- Maîtriser les structures de données fondamentales (chaînes, listes, tuples, dictionnaires).
- Écrire des scripts structurés utilisant variables, fonctions, modules et exceptions.
- Mettre en œuvre les principes de base de la programmation orientée objet en Python.
- Se préparer efficacement à l'examen PCAP grâce aux entraînements et au voucher inclus.

PUBLIC CIBLE

- Débutants en programmation souhaitant se spécialiser en Python.
- Étudiants, développeurs juniors ou profils en reconversion vers les métiers du développement.
- Ingénieurs, techniciens, analystes de données souhaitant consolider leurs bases en Python.

PRÉREQUIS

- Aucun prérequis avancé en programmation n'est obligatoire.
- Être à l'aise avec l'utilisation d'un ordinateur et d'Internet.
- Une première exposition à la logique algorithmique constitue un plus.

PROGRAMME DE LA FORMATION – DÉTAILLÉ

Introduction à Python et à l'environnement de développement

Découvrir le langage Python

- Historique, usages et écosystème de Python (web, data, scripts, automatisation…).
- Installation de Python, IDE et éditeurs (IDLE, VS Code, autres).
- Utiliser l'interpréteur, le shell Python et exécuter des scripts.

Premiers pas avec la syntaxe Python

- Structure d'un script Python, indentation, commentaires.
- Notion d'expression, d'instruction et de bloc de code.

Variables, types de données et opérateurs

Types fondamentaux et conversions

- Entiers, flottants, booléens, chaînes de caractères.
- Conversions de types (casting) et fonctions intégrées (int, float, str, bool…).

Variables et opérateurs

- Déclaration et affectation, conventions de nommage.
- Opérateurs arithmétiques, de comparaison et logiques.
- Priorité des opérateurs et parenthèses.

Entrées / sorties simples

- Afficher des résultats avec print().
- Lire des données utilisateur avec input() et les convertir.

Contrôle de flux : conditions et boucles

Instructions conditionnelles

- if, elif, else : structure, bonnes pratiques, imbrication.
- Conditions composites avec and, or, not.

Boucles et itérations

- Boucle while : structure, cas d'usage, gestion de fin de boucle.
- Boucle for : itération sur séquences et objets itérables.
- break, continue, else sur les boucles.

PROGRAMME DE LA FORMATION – DÉTAILLÉ

Compréhensions de listes (introduction)

- Syntaxe de base des list comprehensions.
- Exemples pratiques pour transformer et filtrer des données.

Structures de données : chaînes, listes, tuples, dictionnaires

Chaînes de caractères

- Indexation, slicing, concaténation, répétition.
- Méthodes courantes (upper, lower, strip, split, join…).
- f-strings et formats de sortie.

Listes et tuples

- Créer, lire, modifier et supprimer des éléments de liste.
- Méthodes de liste : append, insert, remove, pop, sort, reverse…
- Utilisation des tuples et différences avec les listes.

Dictionnaires et ensembles (sets)

- Créer et manipuler des dictionnaires : clés, valeurs, items().
- Parcourir un dictionnaire, ajouter, modifier, supprimer des entrées.
- Notion d'ensembles (set), opérations d'union, intersection, différence.

Fonctions, modules et organisation du code

Définir et utiliser des fonctions

- Définition avec def, paramètres, valeur de retour.
- Arguments positionnels, nommés, valeurs par défaut.
- Portée des variables (locale, globale), mot-clé global.

Modules et packages

- Importer un module, utiliser des fonctions intégrées (math, random, sys…).
- Créer ses propres modules et les réutiliser.

Bonnes pratiques d'organisation

- Structurer un projet simple en fichiers et dossiers.
- Utiliser la clause if __name__ == "__main__".

PROGRAMME DE LA FORMATION – DÉTAILLÉ

Structures de données : chaînes, listes, tuples, dictionnaires

Chaînes de caractères

- Indexation, slicing, concaténation, répétition.
- Méthodes courantes (upper, lower, strip, split, join…).
- f-strings et formats de sortie.

Gestion des erreurs, exceptions et opérations sur les fichiers

Exceptions en Python

- Comprendre les erreurs d'exécution et les exceptions courantes.
- Utiliser try, except, else, finally pour sécuriser le code.
- Lancer ses propres exceptions avec raise.

Travail avec les fichiers

- Ouvrir, lire et écrire dans des fichiers texte.
- Utiliser le mot-clé with pour gérer les ressources proprement.

Introduction à la programmation orientée objet (POO) en Python

Concepts de base de la POO

- Classes, objets, attributs et méthodes.
- Constructeur __init__ et destruction d'objets (notion).

Héritage et encapsulation (niveau PCAP)

- Créer une classe dérivée à partir d'une classe de base.
- Accès aux attributs, méthodes spéciales simples (__str__, etc.).

Préparation à l'examen PCAP & atelier pratique

Objectifs et structure de l'examen PCAP

- Format de l'examen, domaines de compétences, score de réussite.
- Répartition des thèmes : syntaxe, types, structures de données, fonctions, POO, exceptions.

Exercices, quiz et examen blanc

- Résolution guidée d'exercices similaires à ceux de l'examen.
- Simulation d'examen avec un test blanc chronométré.
- Analyse des résultats, conseils personnalisés pour le jour J.