Date de publication : 15/03/2024

Formation : Développer en langage Python orienté objet, certification API Society

Formation pratique - 4j - 28h00 - Réf. YPT
Prix : 3090 CHF H.T.

4,8 / 5

Python est un langage de programmation largement utilisé dans les applications web, le développement logiciel, la science des données, la finance, la cartographie, l'intelligence artificielle et l'apprentissage automatique. Cette formation vous apprendra la syntaxe de base, les différents types de données, les structures conditionnelles et répétitives, les fonctions, les modules, les paquets, la manipulation des fichiers et la gestion des exceptions.

Objectifs pédagogiques

À l'issue de la formation, le participant sera en mesure de :

- Connaître la syntaxe du langage Python
- Connaître et manipuler les types de données fondamentaux
- Utiliser et définir des fonctions, des modules, des paquets, des exceptions, manipuler des fichiers
- Connaître la théorie de la Programmation orientée objet et sa mise en pratique en Python
- Connaître et utiliser les modules incontournables de la bibliothèque standard

Public concerné

Développeurs, ingénieurs, chefs de projet proches du développement.

Prérequis

Connaissance d'un langage de programmation.

Vérifiez que vous avez les prérequis nécessaires pour profiter pleinement de cette formation en faisant ce test.

Certification

L'examen de certification se déroule en ligne, en différé et en français dans le mois qui suit la formation. Il se compose d'une épreuve théorique d'une durée de 20 minutes - 40 questions type QCM Vrai/Faux et informations à saisir, et d'une épreuve pratique de programmation (exercice de code).

PARTICIPANTS

Développeurs, ingénieurs, chefs de projet proches du développement.

PRÉREQUIS

Connaissance d'un langage de programmation.

COMPÉTENCES DU FORMATEUR

Les experts qui animent la formation sont des spécialistes des matières abordées. Ils ont été validés par nos équipes pédagogiques tant sur le plan des connaissances métiers que sur celui de la pédagogie, et ce pour chaque cours qu'ils enseignent. Ils ont au minimum cinq à dix années d'expérience dans leur domaine et occupent ou ont occupé des postes à responsabilité en entreprise.

MODALITÉS D'ÉVALUATION

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques...
Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

Méthodes et moyens pédagogiques

Travaux pratiques

Travaux pratiques individuels et en groupe, réflexion collective.

Méthodes pédagogiques

Pédagogie active favorisant l'implication personnelle et les échanges entre participants.

Modalités d'évaluation

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques…

Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

Programme de la formation

1 Introduction

- Historique (auteur, date de la première version).
- Versions de Python (branches 2 et 3).
- Caractéristiques du langage (multi-paradigme, typage dynamique fort, syntaxe claire).
- Panorama de la bibliothèque standard.
- Modules d'extension et commande pip.
- Principe de fonctionnement de l'interpréteur (bytecode PYC).
- Interpréteur officiel CPython et autres interpréteurs (micropython, brython, pypy, numba.
- Ressources (site Internet python.org, accès aux documentations).
- Fonction help() et chaînes documentaires.
- Principe de l'indentation pour délimiter les blocs d'instruction.
- Commentaire.
- Mots-clés réservés.
- Conventions de nommage.
- Interpréteur interactif.
- Programme autonome.
- Fonctions intégrées élémentaires : print(), type(), input(), len().

MOYENS PÉDAGOGIQUES ET TECHNIQUES

• Les moyens pédagogiques et les méthodes d'enseignement utilisés sont principalement : aides audiovisuelles, documentation et support de cours, exercices pratiques d'application et corrigés des exercices pour les formations pratiques, études de cas ou présentation de cas réels pour les séminaires de formation.
• À l'issue de chaque formation ou séminaire, ORSYS fournit aux participants un questionnaire d'évaluation du cours qui est ensuite analysé par nos équipes pédagogiques.
• Une feuille d'émargement par demi-journée de présence est fournie en fin de formation ainsi qu'une attestation de fin de formation si le participant a bien assisté à la totalité de la session.

MODALITÉS ET DÉLAIS D'ACCÈS

L'inscription doit être finalisée 24 heures avant le début de la formation.

ACCESSIBILITÉ AUX PERSONNES HANDICAPÉES

Pour toute question ou besoin relatif à l'accessibilité, vous pouvez joindre notre équipe PSH par e-mail à l'adresse psh-accueil@orsys.fr.

2 Types de données non modifiables

- Utilité des types non-modifiables (optimisation mémoire), fonctions id() et hash(), opérateur is.
- Principe des séquences ordonnées (str, tuple et list) et collections (dict, set).
- Booléen (bool), objets True et False.
- Nombre (int, float, complex), constructeurs, opérateurs >>, <<, |, &, ^, //, % et **.
- Notations exponentielle, binaire, octale et hexadécimale, fonctions hex(), oct(), bin(), chr(), ord().
- Chaîne de caractères unicode (str), définition avec simple et double guillemets, chaines multilignes, mode raw.
- Indiçage positif et négatif, tranche de valeurs (slice), opérateurs +, * et in, itération.
- Méthodes incontournables de str : split(), replace(), lower(), upper(), strip(), join().
- Chaîne de caractères formatée (%s, %d, %f), méthode format() et f-string.
- Principe du depacking de variables.
- Tableau d'octets (bytes), constructeur.
- Tuple (tuple), constructeur, indiçage, itération, opérateurs +, * et in, méthodes count() et index().
- Objet None et fonction repr().

3 Types de données modifiables

- Listes (list), constructeur, indiçage, intération, opérateurs +, * et in.
- Méthodes append(), insert(), fonction del(), méthodes sort(), reverse(), remove(), extend(), pop(), clear().
- Manipulation de pointeurs, copie superficielle via la méthode copy() ou l'indiçage [:].
- Etude de copie en profondeur avec la fonction deepcopy() du module copy.
- Fonction sorted.
- Principe de fonctionnement des objets itérables.
- Fonctions reversed() et range().
- Dictionnaires (dict), constructeur, indiçage, opérateur in, fonction del().
- Manipulation de pointeurs, copie superficielle via la méthode copy();
- Analyse de copie en profondeur avec la fonction deepcopy() du module copy.
- Set (set), constructeur, opérateurs - | & et ^.

4 Structures conditionnelles et répétitives

- Structure conditionnelle if ... elif ... else.
- Opérateur ternaire et morse.
- Structure répétitive while.
- Structure répétitive for.
- Instructions break et continue.
- Fonction enumerate.
- Bloc else sur structure répétitive.
- Liste en intension (comprehension list), dictionnaire en intension (comprehension dict).

5 Fonctions, modules et paquets

- Définition et appel d'une fonction.
- Espace de noms local, global, prédéfini (__builtins__) et fonction dir().
- Retourner des valeurs, , instruction return.
- Fonctions génériques (duck typing).
- Valeurs par défaut.
- Passage par étiquette.
- Nombre d'arguments arbitraire (*args, **kwargs).
- Fonctions anonymes (lambda).
- Fonctions eval(), exec(), map() et filter().
- Importation de modules.
- Création d'un module.
- Bloc if __name__ == ''__main__''.
- Importation de paquets.
- Création d'un paquet (__init__.py).
- Instruction yield.

6 Manipulation des fichiers

- Fonction open() et méthode close().
- Méthodes readline() et readlines().
- Objet itérable.
- Instruction with avec les fichiers.
- Méthodes read() et write().
- Méthodes tell() et seek().
- Méthode writelines().
- Modules complémentaires : struct, csv, json, xml.
- Sérialisation avec le module pickle.
- Sérialisation avec le module shelve.

7 Programmation Orientée Objet

- Concepts fondamentaux de la POO (séparation du code, encapsulation, héritage).
- Notions de classe d'objet, d'objet (instance), d'attribut et de méthode.
- Définition d'une classe d'objet.
- Instanciation d'objets, fonction isinstance().
- Constructeur (__init__).
- Attributs et méthodes.
- Le paramètre self.
- Surcharge d'affichage (__str__).
- Surcharge d'opérateurs (__eq__, __add__).
- Propriété (fonction spéciale property), accesseur et mutateur.
- Espaces de noms global, de l'objet, de la classe.
- Variable de classe.
- Constructeur à nombre d'arguments arbitraire (*args, **kwargs).
- Agrégation / Composition.
- Héritage de classe (généralisation), fonctions issubclass(), super() et méthode mro().

8 Exceptions

- Principe de fonctionnement.
- Exceptions prédéfinies et arbre d'héritage.
- Instructions try ... except ... else ... finally.
- Propagation des exceptions.
- Déclenchement d'exceptions.
- Définition d'une exception.

9 Modules de la bibliothèque standard

- Interaction avec l'interpréteur : module sys.
- Interaction avec le système d'exploitation : modules os et pathlib..
- Interaction avec le système de fichiers : module os.path.
- Expressions rationnelles : module re, fonctions search(), match(), split() et sub().
- Tests unitaires : instruction assert, module unittest.
- Tour d'horizon de modules de la bibliothèque standard :datetime, math, timeit, urllib, collections, csv, json, sqlite3.
- Présentation de datetime, subprocess, shutil, collections, timeit, urllib, sqlite3...

Dates et lieux

CLASSE À DISTANCE
2026 : 16 juin, 29 sep., 15 déc.