Python, programmation Objet
certification TOSA® à distance en option

Cours Pratique de 5 jours - 35h
Réf : PYT - Prix 2024 : 2 960CHF HT

Python est un langage de programmation multiplateforme permettant le développement d'une grande variété d'applications. Vous en maîtriserez sa syntaxe, ses principaux mécanismes et son paradigme Objet. Vous découvrirez les fonctionnalités de la bibliothèque de modules standards, implémenterez des interfaces graphiques, accéderez aux données d'une base tout en utilisant des outils permettant de tester et d'évaluer la qualité du code produit.

OBJECTIFS PÉDAGOGIQUES

À l'issue de la formation l'apprenant sera en mesure de :

Maîtriser la syntaxe du langage Python

Acquérir les notions essentielles de la programmation objet

Connaître et mettre en œuvre les différents modules Python

Concevoir des interfaces graphiques

Mettre en œuvre les outils de test et d'évaluation de la qualité d'un programme Python

LE PROGRAMME

dernière mise à jour : 10/2023

1) Syntaxe du langage Python

- Les identifiants et les références. Les conventions de codage et les règles de nommage.
- Les blocs, les commentaires.
- Les types de données disponibles.
- Les variables, l'affichage formaté, la portée locale et globale.
- La manipulation des types numériques, la manipulation de chaînes de caractères.
- La manipulation des tableaux dynamiques (liste), des tableaux statiques (tuple) et des dictionnaires.
- L'utilisation des fichiers.
- La structure conditionnelle if/elif/else.
- Les opérateurs logiques et les opérateurs de comparaison.
- Les boucles d'itérations while et for. Interruption d'itérations break/continue.
- La fonction range.
- L'écriture et la documentation de fonctions.
- Les lambda expression.
- Les générateurs.
- La structuration du code en modules.

Travaux pratiques : Installation et prise en main de l'interpréteur Python.

2) Approche Orientée Objet

- Les principes du paradigme Objet.
- La définition d'un objet (état, comportement, identité).
- La notion de classe, d'attributs et de méthodes.
- L'encapsulation des données.
- La communication entre les objets.

PARTICIPANTS

Développeurs, ingénieurs, chefs de projets proches du développement.

PRÉREQUIS

Connaissances de base en 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.

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 stages pratiques, études de cas ou présentation de cas réels pour les séminaires de formation.
• À l'issue de chaque stage 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 stagiaire 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

Vous avez un besoin spécifique d'accessibilité ? Contactez Mme FOSSE, référente handicap, à l'adresse suivante psh-accueil@orsys.fr pour étudier au mieux votre demande et sa faisabilité.

- L'héritage, transmission des caractéristiques d'une classe.
- La notion de polymorphisme.
- Association entre classes.
- Les interfaces.
- Présentation d'UML.
- Les diagrammes de classes, de séquences, d'activités...
- Notion de modèle de conception (Design Pattern).

Travaux pratiques : Modélisation en UML d'un cas d'étude simple.

3) Programmation Objet en Python

- Les particularités du modèle Objet de Python.
- L'écriture de classes et leur instanciation.
- Les constructeurs et les destructeurs.
- La protection d'accès des attributs et des méthodes.
- La nécessité du paramètre Self.
- L'héritage simple, l'héritage multiple, le polymorphisme.
- Les notions de visibilités.
- Les méthodes spéciales.
- L'introspection.
- L'implémentation des interfaces.
- Les bonnes pratiques et les modèles de conception courants.
- L'utilisation du mécanisme d'exception pour la gestion des erreurs.

Travaux pratiques : Pratique des différents concepts Objet au travers de l'implantation de l'étude de cas.

4) Utilisation StdLib

- Les arguments passés sur la ligne de commande.
- L'utilisation du moteur d'expressions régulières Python avec le module "re", les caractères spéciaux, les cardinalités.
- La manipulation du système de fichiers.
- Présentation de quelques modules importants de la bibliothèque standard : module "sys", "os", "os.path".
- Empaquetage et installation d'une bibliothèque Python.
- Les accès aux bases de données relationnelles, le fonctionnement de la DB API.

Travaux pratiques : Mise en œuvre de modules Python : expressions régulières, accès à une base de données,

5) Outils QA

- Les outils d'analyse statique de code (Pylint, Pychecker).
- L'analyse des comptes rendus d'analyse (types de messages, avertissements, erreurs).
- Extraction automatique de documentation.
- Le débogueur de Python (exécution pas à pas et analyse post-mortem).
- Le développement piloté par les tests.
- Les modules de tests unitaires Python (Unittest...).
- L'automatisation des tests, l'agrégation de tests.
- Les tests de couverture de code, profiling.

Travaux pratiques : Utilisation des outils pylint et pychecker pour la vérification d'un code Python. Mise en œuvre de tests unitaires.

6) Création IHM TkInter

- Les principes de programmation des interfaces graphiques.
- Présentation de la bibliothèque TkInter.
- Les principaux conteneurs.
- Présentation des widgets disponibles (Button, Radiobutton, Entry, Label, Listbox, Canvas, Menu, Scrollbar, Text...).
- Le gestionnaire de fenêtres.
- Le placement des composants, les différents layouts.

- La gestion des événements, l'objet event.
- Les applications multifenêtres.
Travaux pratiques : Conception d'une interface graphique avec la bibliothèque Tkinter.

7) Interfaçage Python/C
- Présentation du module Ctypes.
- Le chargement d'une librairie C.
- Appel d'une fonction.
- La réécriture d'une fonction Python en C avec l'API Python/C.
- La création de modules C pour Python.
- L'interpréteur Python dans C.
- L'utilisation du profileur de code.
Travaux pratiques : Appel de fonctions écrites en C depuis Python. Création de modules C pour Python.

8) Conclusion
- Analyse critique de Python.
- L'évolution du langage.
- Eléments de webographie et de bibliographie.

LES DATES

CLASSE À DISTANCE
2024 : 13 mai, 03 juin, 17 juin, 01 juil., 02 sept., 16 sept., 30 sept., 14 oct., 04 nov., 18 nov., 02 déc., 09 déc.

GENÈVE
2024 : 30 sept., 09 déc.

LAUSANNE
2024 : 30 sept., 09 déc.