

Intitulé de la certification

Développer en langage Python orienté objet

Description des métiers

Cette certification s'adresse aux développeurs informatiques en tant que compétence complémentaire à leur métier, et aux autres métiers pour lesquels le langage Python est devenu une compétence transversale : analyste de données, biostatisticien, cartographe, océanologue, trader, chercheur ...

Référentiel d'activités, de compétences et d'évaluation

Référentiel d'activités	Référentiel de compétences	Référentiel d'évaluation	
		Modalités d'évaluation	Critères d'évaluation
<p>décrit les situations de travail et les activités exercées, les métiers ou emplois visés</p>	<p>identifie les connaissances et les compétences, y compris transversales, qui découlent du référentiel d'activités</p>	<p>définit les critères et les modalités d'évaluation des acquis</p>	
<p>A1 - Écrire un programme Python structuré en fonctions</p> <p>(emploi / métier : exemple de situation de travail)</p> <p>Biostatisticien : simuler un modèle prédictif sur un jeu de données</p>	<p>C1 - Maitriser la syntaxe et la sémantique de Python afin d'écrire des programmes fonctionnels</p> <p>C2 - Connaître les types de données fondamentaux (nombre, chaîne de caractères, tuple, liste et dictionnaire) et les méthodes associées afin de ne pas ré-écrire des fonctionnalités déjà fournies par Python</p>	<p>Épreuve théorique : le candidat doit répondre à un questionnaire (QCU, QCM et informations à saisir) de 40 questions tirées aléatoirement parmi une banque de plus de 200 questions</p> <p>Durée totale de l'épreuve théorique : 20 minutes</p>	<p>Épreuve théorique : le candidat a répondu 70% de bonnes réponses ou plus au questionnaire, soit plus de 28 réponses justes sur 40 questions posées</p>

<p><u>Cartographe</u> : écrire une extension au logiciel de cartographie QGIS</p> <p><u>Administrateur système</u> : écrire un programme de supervision</p> <p>...</p>	<p>C3 - Choisir le bon type en fonction de l'information à manipuler et du traitement à réaliser afin de traiter efficacement et de manière claire des grands volumes de données</p> <p>C4 - Connaître les fonctions intégrées de Python (sort, len, abs, chr, ord ...) afin d'accéder aux fonctionnalités associées</p> <p>C5 - Maitriser les structures de contrôle conditionnelles et répétitives afin d'automatiser un traitement de données</p> <p>C6 - Utiliser l'imbrication des différentes structures de contrôle en vue d'appliquer un traitement à un jeu de données</p> <p>C7 - Créer des fonctions spécifiques utilisant les fonctions et les classes élémentaires génériques fournies par le langage</p> <p>C11 - Connaître le principe des espaces de noms (global, local, pré-défini) afin de comprendre la portée des variables et de segmenter la mémoire en vue d'écrire des programmes sécurisés</p> <p>C12 - Documenter l'utilité et l'usage d'une fonction, d'un module et d'un paquet en vue d'améliorer sa qualité et sa ré-utilisabilité</p> <p>C21 - Transposer en langage Python un algorithme décrit dans un cahier des charges afin de mettre en oeuvre le besoin exprimé</p> <p>C22 - Mettre en oeuvre les bonnes structures de données et écrire les fonctions adéquates pour concevoir des algorithmes d'une certaine complexité</p>	<p>Épreuve pratique : le candidat doit écrire un programme contenant une fonction mettant en oeuvre des structures conditionnelles et répétitives imbriquées manipulant des données structurées (chaîne, liste, dictionnaire ...)</p> <p>Durée totale de l'épreuve pratique : 2 heures</p> <p>Temps alloué à cet exercice : 30 minutes</p>	<p>Épreuve pratique : le travail rendu par le candidat est corrigé par un membre du jury professionnel développeur Python en activité selon les critères suivants :</p> <p>CR1 - le candidat a écrit un programme qui contient une fonction</p> <p>CR2 - le candidat a écrit un programme avec une structure répétitive contenant une structure conditionnelle</p> <p>CR3 - le candidat a choisi la structure de données adaptée</p> <p>CR4 - le candidat connaît et utilise les méthodes fournies par les types de données fondamentaux</p>
<p>A2 - Importer et utiliser un module existant</p> <p>(emploi / métier : exemple de situation de travail)</p> <p><u>Ingénieur en Intelligence Artificielle</u> : utiliser le module Scikit-learn pour faire de l'apprentissage automatique</p> <p><u>Data Analyst</u> : utiliser le module Matplotlib pour dessiner les variations d'un cours boursier</p>	<p>C8 - Importer des modules existants et exploiter les fonctions et classes incluses</p> <p>C11 - Connaître le principe des espaces de noms (global, local, pré-défini) afin de comprendre la portée des variables et de segmenter la mémoire en vue d'écrire des programmes sécurisés</p> <p>C19 - Interagir avec l'interpréteur Python ou avec le système de fichiers (modules sys, os, os.path)</p> <p>C20 - Ecrire des expressions rationnelles (expressions régulières) afin de réaliser des recherches de correspondances, des</p>	<p>Épreuve théorique : le candidat doit répondre à un questionnaire (QCU, QCM et informations à saisir) de 40 questions tirées aléatoirement parmi une banque de plus de 200 questions</p> <p>Durée totale de l'épreuve théorique : 20 minutes</p> <p>Épreuve pratique : le candidat doit importer un module existant de la bibliothèque standard, afficher sa documentation et utiliser des fonctions de ce module</p>	<p>Épreuve théorique : le candidat a répondu 70% de bonnes réponses ou plus au questionnaire, soit plus de 28 réponses justes sur 40 questions posées</p> <p>Épreuve pratique : le travail rendu par le candidat est corrigé par un membre du jury professionnel développeur Python en activité selon les critères suivants :</p>

<p><u>Océanologue</u> : utiliser le module Pandas pour analyser des données issues de balises marines</p> <p>...</p>	<p>remplacements, des extractions très sophistiquées sur les chaînes de caractères</p> <p>C23 - Choisir les modules Python pertinents et savoir en mesurer les performances pour écrire des programmes manipulant rapidement de grands volumes de données</p>	<p>Durée totale de l'épreuve pratique : 2 heures</p> <p>Temps alloué à cet exercice : 15 minutes</p>	<p>CR5 - le candidat sait importer un module existant en utilisant les deux techniques possibles</p> <p>CR6 - le candidat sait afficher la documentation d'un module</p>
<p>A3 - Écrire un module Python en vue de le réutiliser dans d'autres programmes et le partager avec d'autres développeurs</p> <p>(emploi / métier : exemple de situation de travail)</p> <p><u>Développeur</u> : écrire un module de connexion à une base de données pour qu'il soit utilisé par d'autres développeurs</p>	<p>C9 - Créer son propre module Python afin d'écrire du code ré-utilisable depuis un autre programme</p> <p>C11 - Connaître le principe des espaces de noms (global, local, pré-défini) afin de comprendre la portée des variables et de segmenter la mémoire en vue d'écrire des programmes sécurisés</p> <p>C12 - Documenter l'utilité et l'usage d'une fonction, d'un module et d'un paquet en vue d'améliorer sa qualité et sa ré-utilisabilité</p>	<p>Épreuve théorique : le candidat doit répondre à un questionnaire (QCU, QCM et informations à saisir) de 40 questions tirées aléatoirement parmi une banque de plus de 200 questions</p> <p>Durée totale de l'épreuve théorique : 20 minutes</p>	<p>Épreuve théorique : le candidat a répondu 70% de bonnes réponses ou plus au questionnaire, soit plus de 28 réponses justes sur 40 questions posées</p>
<p>A4 - Créer un paquet Python pour structurer un logiciel complexe</p> <p>(emploi / métier : exemple de situation de travail)</p> <p><u>Développeur</u> : créer un paquet contenant plusieurs modules pour rendre possible le développement à plusieurs programmeurs</p>	<p>C10 - Créer son propre paquet Python afin de structurer plusieurs modules dans une hiérarchie de répertoires et d'espaces de noms</p> <p>C11 - Connaître le principe des espaces de noms (global, local, pré-défini) afin de comprendre la portée des variables et de segmenter la mémoire en vue d'écrire des programmes sécurisés</p> <p>C12 - Documenter l'utilité et l'usage d'une fonction, d'un module et d'un paquet en vue d'améliorer sa qualité et sa ré-utilisabilité</p>	<p>Épreuve théorique : le candidat doit répondre à un questionnaire (QCU, QCM et informations à saisir) de 40 questions tirées aléatoirement parmi une banque de plus de 200 questions</p> <p>Durée totale de l'épreuve théorique : 20 minutes</p>	<p>Épreuve théorique : le candidat a répondu 70% de bonnes réponses ou plus au questionnaire, soit plus de 28 réponses justes sur 40 questions posées</p>
<p>A5 - Ouvrir un fichier, lire son contenu, écrire un fichier</p> <p>(emploi / métier : exemple de situation de travail)</p> <p><u>BIM Manager</u> : générer un fichier XML à partir d'informations extraites d'une base de données et de fichiers</p> <p><u>Data Analyst</u> : ouvrir un fichier CSV, appliquer un traitement sur certaines lignes, et écrire un nouveau fichier CSV avec les modifications</p>	<p>C13 - Lire et écrire un fichier en format texte ou en format binaire afin d'appliquer un traitement sur les données lues ou écrites</p>	<p>Épreuve théorique : le candidat doit répondre à un questionnaire (QCU, QCM et informations à saisir) de 40 questions tirées aléatoirement parmi une banque de plus de 200 questions</p> <p>Durée totale de l'épreuve théorique : 20 minutes</p> <p>Épreuve pratique : le candidat doit créer un programme qui ouvre un fichier texte, affiche son contenu, et le ferme proprement</p>	<p>Épreuve théorique : le candidat a répondu 70% de bonnes réponses ou plus au questionnaire, soit plus de 28 réponses justes sur 40 questions posées</p> <p>Épreuve pratique : le travail rendu par le candidat est corrigé par un membre du jury professionnel développeur Python en activité selon les critères suivants :</p>

<p>...</p>		<p>Durée totale de l'épreuve pratique : 2 heures</p> <p>Temps alloué à cet exercice : 15 minutes</p>	<p>CR7 - le candidat a écrit un programme qui ouvre correctement le fichier</p> <p>CR8 - le candidat a écrit un programme qui affiche le contenu du fichier</p> <p>CR9 - le candidat a écrit un programme qui ferme proprement le fichier</p>
<p>A6 - Écrire un programme orienté objet</p> <p>(emploi / métier : exemple de situation de travail)</p> <p>Développeur : écrire un progiciel de gestion intégrant la gestion comptable et financière, les RH, la vente, la distribution, l'approvisionnement ...</p> <p>Data Scientist : industrialiser dans une classe métier les petits programmes écrits par des Data Analysts pour les intégrer au système d'information</p> <p>...</p>	<p>C14 - Connaître les grands principes de la programmation orientée objet (encapsulation, héritage ...) et les notions associées (constructeur, attributs, méthodes ...) en vue de les mettre en oeuvre en Python afin d'écrire des programmes modulaires et évolutifs</p> <p>C15 - Définir ses propres classes d'objet en vue de créer des objets correspondants parfaitement aux données manipulées</p> <p>C16 - Documenter une classe et ses méthodes en vue d'améliorer sa qualité et sa ré-utilisabilité</p> <p>C17 - Intercepter une exception (erreur) qui s'est produite en vue de la gérer pour que le programme continue de fonctionner normalement</p> <p>C18 - Lever une exception afin de remonter une erreur via le mécanisme de propagation des exceptions et pouvoir ainsi la gérer à un endroit différent du programme</p> <p>C21 - Transposer en langage Python un algorithme décrit dans un cahier des charges afin de mettre en oeuvre le besoin exprimé</p> <p>C22 - Mettre en oeuvre les bonnes structures de données et écrire les fonctions adéquates pour concevoir des algorithmes d'une certaine complexité</p>	<p>Épreuve théorique : le candidat doit répondre à un questionnaire (QCU, QCM et informations à saisir) de 40 questions tirées aléatoirement parmi une banque de plus de 200 questions</p> <p>Durée totale de l'épreuve théorique : 20 minutes</p> <p>Épreuve pratique : le candidat doit écrire une classe objet, définir des méthodes et les appeler depuis le programme principal</p> <p>Durée totale de l'épreuve pratique : 2 heures</p> <p>Temps alloué à cet exercice : 30 minutes</p>	<p>Épreuve théorique : le candidat a répondu 70% de bonnes réponses ou plus au questionnaire, soit plus de 28 réponses justes sur 40 questions posées</p> <p>Épreuve pratique : le travail rendu par le candidat est corrigé par un membre du jury professionnel développeur Python en activité selon les critères suivants :</p> <p>CR10 - le candidat a écrit une classe objet contenant un constructeur qui initialise les attributs de l'objet</p> <p>CR11 - le candidat a écrit une classe objet contenant les méthodes demandées par l'exercice</p> <p>CR12 - le candidat a écrit le programme principal qui créé un objet (une instance) de la classe</p> <p>CR13 - le candidat a écrit le programme principal qui appelle correctement les méthodes sur l'objet</p>
<p>A7 - Réaliser des traitements complexes sur les chaînes de caractères</p> <p>(emploi / métier : exemple de situation de travail)</p>	<p>C20 - Ecrire des expressions rationnelles (expressions régulières) afin de réaliser des recherches de correspondances, des remplacements, des extractions très sophistiquées sur les chaînes de caractères</p>	<p>Épreuve théorique : le candidat doit répondre à un questionnaire (QCU, QCM et informations à saisir) de 40 questions tirées aléatoirement parmi une banque de plus de 200 questions</p>	<p>Épreuve théorique : le candidat a répondu 70% de bonnes réponses ou plus au questionnaire, soit plus de 28 réponses justes sur 40 questions posées</p>

<p>Développeur : contrôler des informations saisies par l'utilisateur du logiciel</p> <p>Chercheur en économie : extraire des informations d'un fichier contenant des informations non structurées</p> <p>Trader : surveiller plusieurs cours boursiers provenant de différentes sources, les analyser afin de passer des ordres d'achat et de vente</p> <p>...</p>	<p>C22 - Mettre en oeuvre les bonnes structures de données et écrire les fonctions adéquates pour concevoir des algorithmes d'une certaine complexité</p>	<p>Durée totale de l'épreuve théorique : 20 minutes</p> <p>Épreuve pratique : le candidat doit réaliser des traitements sur des chaînes de caractères, soit avec les outils intégrés de Python, soit avec le module "re"</p> <p>Durée totale de l'épreuve pratique : 2 heures</p> <p>Temps alloué à cet exercice : 30 minutes</p>	<p>Épreuve pratique : le travail rendu par le candidat est corrigé par un membre du jury professionnel développeur Python en activité selon les critères suivants :</p> <p>CR14 - le candidat a écrit un programme qui réalise les traitements sur les chaînes de caractères demandés par l'exercice</p> <p>CR15 - La solution proposée par le candidat est simple et pertinente</p> <p>CR16 - Le candidat a réussi à décomposer un problème complexe en plusieurs petits problèmes simples faciles à résoudre</p>
<p>A8 - Écrire un programme respectant les règles de nommage Python et intégrant des tests unitaires</p> <p>(emploi / métier : exemple de situation de travail)</p> <p>Développeur : travailler avec d'autres développeurs sur un logiciel complexe qui va évoluer sur plusieurs années</p>	<p>C24 - Respecter les règles de nommage Python décrites dans le PEP0008 et savoir utiliser un analyse syntaxique de code afin d'écrire des programmes de qualité lisibles par d'autres développeurs</p> <p>C25 - Identifier le code pertinent à tester et écrire des tests unitaires afin de réaliser des programmes stables et durables</p>	<p>Épreuve théorique : le candidat doit répondre à un questionnaire (QCU, QCM et informations à saisir) de 40 questions tirées aléatoirement parmi une banque de plus de 200 questions</p> <p>Durée totale de l'épreuve théorique : 20 minutes</p>	<p>Épreuve théorique : le candidat a répondu 70% de bonnes réponses ou plus au questionnaire, soit plus de 28 réponses justes sur 40 questions posées</p>