TOSA

Vos compétences. Votre différence.

Référentiel pédagogique

Python

contact@isograd.com

TOSA code

Table des matières

Introduction au référentiel pédagogique 3
- Le Tosa® 4
- Objet du référentiel pédagogique 4
- Une échelle de score unique 4
- Domaines et sous-domaines de compétences 6

Niveau 1 - Initial 7
- Synthèse 8

Niveau 2 - Basique 9
- Langage et syntaxe 10
- Structures de données et objets 10
- Modules et packages 10
- Optimisation de code 11
- Synthèse 12

Niveau 3 - Opérationnel 13
- Langage et syntaxe 14
- Structures de données et objets 14
- Modules et packages 14
- Optimisation de code 15
- Synthèse 16

Niveau 4 - Avancé 18
- Langage et syntaxe 19
- Structures de données et objets 19
- Modules et packages 19
- Optimisation de code 20
- Synthèse 21

Niveau 5 - Expert 22
- Langage et syntaxe 23
- Structures de données et objets 23
- Modules et packages 24
- Optimisation de code 24
- Synthèse 25

Introduction au référentiel pédagogique

Pour l'évaluation et la certification Tosa

Le Tosa®

Les tests d'évaluation et de certification Tosa® permettent de déterminer le niveau des compétences et les aptitudes d'un candidat sur les logiciels bureautiques, les outils digitaux et els langages de programmation utilisés dans un environnement professionnel.

Les tests Tosa® sont ainsi conçus pour valider les compétences professionnelles des candidats souhaitant améliorer leur employabilité (salariés, étudiants, demandeurs d'emploi, personnes en reconversion).

Les évaluations et certifications Tosa® sont des tests adaptatifs, élaborés selon des méthodologies scientifiques (la détermination du score est basée sur l'Item Response Theory (IRT)). Elles délivrent ainsi un diagnostic détaillé sur les compétences de chaque candidat.

La robustesse et la fiabilité des tests Tosa® tiennent donc à l'association d'un modèle mathématique d'analyse de la difficulté et de la pertinence des questions (IRT). C'est un modèle très proche de celui utilisé par le GMAT.

Objet du référentiel pédagogique

Ce référentiel pédagogique présente l'ensemble des compétences évaluées dans les domaines et sous-domaines des tests d'évaluation et de certification Tosa® Python.

Il précise les compétences techniques attendues pour chaque niveau, et cela dans chacun des quatre domaines de compétences du langage Python. Il s'agit donc d'un outil d'accompagnement dans l'élaboration de programmes d'enseignement ou de formation adaptés au niveau visé par chaque candidat.

Une échelle de score unique

L'évaluation et la certification Tosa® reposent sur une échelle de score unique, traduite en cinq niveaux :

- D'Initial à Expert, pour l'évaluation ;
- De 1 à 1000 pour la certification.

Niveaux Tosa®	Scores Tosa®
Expert	876 - 1000
Avancé	726 – 875
Opérationnel	551 – 725
Basique	351 – 550
Initial	1 – 350

La certification Tosa Python est délivrée avec indication d'un score (entre 551 et 1000), correspondant à un niveau (Opérationnel, Avancé ou Expert). En deçà du score de 551 points le candidat se verra délivrer une attestation de passage de la certification.

L'évaluation Tosa Python est quant à elle délivrée avec indication d'un niveau allant d'Initial à Expert.

TOSA code

Domaines et sous-domaines de compétences

Langage et syntaxe	▪ Contexte et cas d'usages de Python ▪ Syntaxe et sémantique ▪ Gestion des entrées / sorties
Structures de données et objets	▪ Import et utilisation ▪ Bibliothèque standard
Modules et packages	▪ Structures classiques ▪ Fonctions et procédures ▪ Programmation orientée objet
Optimisation de code	▪ Performance de code ▪ Algorithmique

Niveau 1 - Initial

Entre 1 et 350 points

Le niveau initial pour un test d'évaluation est le niveau le moins élevé sur l'échelle de score Tosa®. Il correspond au niveau d'un candidat qui n'a que très peu utilisé Python ou qui n'a que des notions très parcellaires et limitées du fonctionnement du langage.

L'obtention du niveau initial signifie que le candidat connaît peu, voire pas du tout, les fonctionnalités même simples de Python, et qu'il ne peut l'utiliser dans un environnement professionnel.

Synthèse

Domaines	Compétences
Langage et syntaxe	Créer une variable Ajouter un commentaire Afficher le contenu d'une variable
Structures de données et objets	Reconnaître des objets simples de type intégré (*built-in*) : chaînes de caractère, numériques. Reconnaître et créer des booléens
Modules et packages	Connaître l'intérêt des packages Importer un package à partir du nom
Optimisation de code	Retourner une solution dans un format standardisé

Niveau 2 - Basique

Entre 351 et 550 points

Préalablement à l'acquisition des compétences du niveau Basique, le candidat aura maîtrisé les compétences du niveau Initial

Langage et syntaxe

Mettre en place des contrôles de flux

Utiliser les fonctionnalités de contrôles de flux afin d'automatiser un traitement de données conditionnels et/ou répétitifs.

Application métier : Reconnaissance de cas d'usage typique de Python. Mise en place de processus d'extraction de donnée à partir d'une source connue et stable.

Structures de données et objets

Gérer les objets *built-in* simple

Reconnaître et créer des objets simples de type intégrés (*built-in*), en vue de créer des programmes manipulant des données.

Application métier : Automatisation des traitements de données dont les données d'entrée sont de type basique connu et fixe (ex : nombres, chaînes de caractère)

Modules et packages

Importer des fonctionnalités externes

Maîtriser les concepts liés aux modules en vue d'inclure des fonctionnalités d'un module à un programme.

Application métier : création d'un programme réutilisant des fonctions et objets élémentaires créés par un autre membre de l'entreprise.

Optimisation de code

Traiter automatiquement des données

Stocker et traiter simultanément plusieurs données, en vue de créer un flux de traitement de données simples adapté.

Application métier : Sans structure préalablement créée et à partir uniquement d'une problématique, création d'un programme complet répondant à un cahier des charges.

Synthèse

Domaines	Compétences
Langage et syntaxe	Identifier des types d'objets simples Connaître les principaux cas d'usage de Python Utiliser les opérateurs arithmétiques Créer un contrôle de flux conditionnel Créer des boucles simples
Structures de données et objets	Reconnaître tous les objets structurés de type intégré Appliquer les fonctions min, max et len Créer et manipuler un dictionnaire et un ensemble
Modules et packages	Importer un module spécifique d'un package Importer et utiliser des fonctions d'un module Utiliser ces fonctions au sein d'un programme
Optimisation de code	Lire une donnée en entrée Lire et prétraiter plusieurs données d'entrée Choisir les types de variables adaptés au problème Combiner des types de données différents

Niveau 3 – Opérationnel

Entre 551 et 725 points

Préalablement à l'acquisition des compétences du niveau Opérationnel, le candidat aura maîtrisé les compétences du niveau Basique

Langage et syntaxe

Combiner les boucles et contrôles de flux

Créer un programme pour des traitements complets de données, en utilisant les fonctionnalités avancées des contrôles de flux et des opérations arithmétiques.

Application métier : Agrégation de données disponible à partir d'une extraction existante. Mise en place de métriques pertinentes au besoin spécifique métier.

Structures de données et objets

Manipuler les objets *built-in* structurés

Reconnaître les objets structurés de type intégré (*list, set, dict*), afin d'effectuer des traitements de multiples variables simples.

Application métier : Création d'indicateurs complexes à partir de données simples (ex : suivi mensuel des revenus moyens sur une période donnée).

Créer des objets et fonctions adaptées à un problème

Manipuler les objets intégrés simples et structurés et les assembler dans le but de créer des fonctions simples et réutilisables.

Application métier : Création de fonctions performant des tâches simples afin de résoudre plus rapidement un même problème se présentant plusieurs fois.

Modules et packages

Sélectionner et utiliser des fonctionnalités de modules externes

Sélectionner et importer des fonctions spécifiques d'un package de l'API afin de les réutiliser dans un traitement de données.

Application métier : intégration d'un outil spécifique de la bibliothèque standard afin de résoudre un problème classique (traitement de date, fonction mathématiques simples par exemple).

Créer un package

Créer un package simple complet pour créer des fonctionnalités partageables et réutilisables.

Application métier : à partir de fonctions et objets préalablement créés, les regrouper dans une package afin qu'ils puissent être réutilisés de manière cohérente.

Optimisation de code

Manipuler un volume important de données

Modéliser un problème et automatiser des manipulations de données, afin de traiter un volume important ou complexe de données.

Application métier : Mobilisation des premiers concepts d'algorithmique afin d'accélérer le traitement de grand volume de données, qui serait trop lent pour être utile en entreprise si résolu naïvement.

Synthèse

Domaines	Compétences
Langage et syntaxe	Connaître les différentes versions de Python Maîtriser l'ordre des opérations Créer les variables nécessaires Maîtriser les instructions de contrôle de flux associées aux boucles Lire des données sur l'entrée standard
Structures de données et objets	Documenter une fonction Effectuer une itération sur un objet de type structuré Manipuler les fonctions associées aux objets structurés Créer des fonctions
Modules et packages	Connaître et utiliser les packages math et random Créer un module Distinguer les composants internes des modules Distinguer les principaux packages de la bibliothèque standard

Optimisation de code	• Trier des données • Modéliser un graphe et l'explorer • Créer des combinaisons de boucles et conditions adaptées à différents problèmes • Extraire automatiquement une information d'un volume important de données

Niveau 4 - Avancé

Entre 726 et 875 points

Préalablement à l'acquisition des compétences du niveau Avancé, le candidat aura maîtrisé les compétences du niveau Opérationnel

Langage et syntaxe

Maîtriser complètement la syntaxe

Connaître les règles syntaxiques et sémantiques de Python afin de créer des programmes clairs et réutilisables.

Application métier (ex : développeur Python) : création de programmes robustes, réutilisables et maintenables par un autre développeur.

Manipuler des fichiers externes

Maîtriser les outils d'entrée-sortie, dans le but de lire et modifier des documents externes au programme.

Application métier : Automatisation de traitement de données à partir de documents externes, et sauvegarde de ce traitement pour un archivage ou une réutilisation future.

Structures de données et objets

Manipuler les objets de types intégrés

Maîtriser les spécificités des types intégrés afin de traiter efficacement et de manière claire des grands volumes de données.

Application métier : Gestioon des grands volumes de données et agrégation de plusieurs éléments (par exemple plusieurs colonnes, pour de la donnée structurée).

Créer des fonctions

Créer des fonctions documentées, des classes et leurs méthodes associées, dans le but de créer des fonctionnalités intégrables dans d'autres programmes.

Application métier : Création d'objets spécifiques, pour poser les bases techniques d'un projet complexe et rendre ses bases les plus maintenables et partageables possible.

Modules et packages

Importer et utiliser des packages

Importer des packages composés pour exploiter l'ensemble des fonctionnalités incluses.

Application métier : mise en place d'outils d'un module donné au sein d'un projet.

Utiliser la bibliothèque standard

Utiliser les packages fondamentaux de la bibliothèque standard afin de manipuler efficacement des données externes habituelles.

Application métier : mise à profit de plusieurs outils d'un même package de la bibliothèque standard afin de résoudre une problématique complexe. Par exemple : générer de la donnée aléatoire spécifique avec random ou résoudre numériquement des expressions avec math

Optimisation de code

Créer des structures adaptées et optimisées

Implémenter des structures de données adaptées et choisir les fonctions adaptées afin de concevoir des programmes à la complexité algorithmique maîtrisée.

Application métier : création de structures de données spécifiques pour répondre le plus efficacement possible à des problématiques impliquant des volumes importants de données complexes. Compréhension de la complexité algorithmique de plusieurs programmes existants (constant, linéaire, quadratique) afin de sélectionner l'option la plus efficiente.

Synthèse

Domaines	Compétences
Langage et syntaxe	Distinguer les spécificités cœur du langage Python Maîtriser les littéraux Connaître les règles syntaxiques sur les variables Utiliser toutes les fonctionnalités d'impression sur les sorties standard et d'erreur Lire et modifier un fichier
Structures de données et objets	Définir des structures par compréhension Choisir le bon type de variable en fonction de la mutabilité Créer des f-strings sur différents types Concevoir des fonctions à différents types d'arguments Créer une classe et son initialisation
Modules et packages	Naviguer dans un package avec la fonction dir Installer des nouveaux packages avec pip Distinguer les cas d'usages de la plupart des packages de la bibliothèque standard Réutiliser des structures de données et fonctions de la bibliothèque standard
Optimisation de code	Créer des structures de données optimisées pour différents problèmes Utiliser les fonctions standards les plus performantes selon les cas Distinguer la complexité algorithmique de programmes simples (constant, linéaire, quadratique)

Niveau 5 - Expert

Entre 876 et 1000 points

Préalablement à l'acquisition des compétences du niveau Expert, le candidat aura maîtrisé les compétences du niveau Avancé

Langage et syntaxe

Gérer les erreurs

Intégrer la gestion d'erreur afin de créer des programmes couvrant tous les cas d'entrée et de gestion de données.

Application métier (ex : Responsable d'équipe technique) : Perfectionnement d'un programme existant pour qu'il gère de façon lisible les erreurs et données inconnues.

À ce niveau, le candidat peut former sur le langage.

Documenter et partager un programme

Utiliser les règles et conventions de nommage afin de créer des programmes pouvant s'intégrer dans un environnement complet.

Application métier : Création de programmes respectant des conventions de nommage et de style s'intégrant dans une base de code commune à l'entreprise, et amélioration de sa maintenabilité.

Structures de données et objets

Produire des fonctionnalités performantes et adaptées

Mettre en œuvre l'intégralité des outils fonctionnels, afin de créer des fonctions adaptées et efficaces.

Application métier : Automatisation des tâches les plus complexes d'un projet, en maintenant un haut niveau de maintenabilité.

Structurer une solution autour d'objets

Utiliser et implémenter les concepts de la programmation orientée objet, afin de créer des structures spécifiques à un projet.

Application métier : Création d'objets complexes répondant à des applications métier spécifiques et pouvant parfaitement s'intégrer à un projet de grande envergure.

Modules et packages

Produire des modules complets partageables

Maîtriser l'ensemble des concepts liés aux modules pour intégrer des packages de la manière la plus performante possible.

Application métier : Exploitation et maintenance de composants techniques, même lorsqu'ils sont mal documentés.

Mettre entièrement à profit la bibliothèque standard

Utiliser l'intégralité de la bibliothèque standard pour traiter des données et documents externes, en local ou sur internet.

Application métier : Création des traitements de données externes de formats variés (csv ou json en particulier), en naviguant sur l'intégralité des fichiers locaux, ou sur internet.

Optimisation de code

Résoudre de façon performante des problèmes complexes

Mettre en place des outils algorithmiques avancés afin de concevoir des programmes utilisant le moins de ressources possibles.

Application métier : Mise en place d'outils algorithmiques avancés afin de drastiquement améliorer la performance d'un programme existant.

À ce niveau, le candidat peut former sur le logiciel.

Synthèse

Domaines	Compétences
Langage et syntaxe	Capter et gérer les erreurs Distinguer et créer différents types d'erreur Intégrer les erreurs aux contrôles de flux Différencier les priorités de scopes et namespaces Distinguer les environnements Python
Structures de données et objets	Utiliser les fonctions lambda Utiliser des décorateurs Créer et d'utiliser des générateurs Définir les méthodes spécifiques à une classe Créer une structure de donnée adaptée à un problème Gérer l'héritage entre les différentes classes
Modules et packages	Reconnaître des fichiers compilés et leur intérêt Importer des fonctions grâce aux références internes Interagir avec internet via un script Traiter des documents de différents formats Utiliser les packages sys et os
Optimisation de code	Analyser et optimiser un code existant Reconnaître un cas d'application et utiliser une structure de données appropriée

TOSA

Vos compétences. Votre différence.

contact@isograd.com

www.tosa.org