Fundamentos de Python 1

Alcance y Secuencia

Versión1.0

Desarrollado en colaboración con
Python Institute Open Education and Development Group

Contenido

Audiencia

El plan de estudios Python Essentials 1 está diseñado para estudiantes que buscan adquirir habilidades fundamentales en Python y en programación de computadoras:

- estudiantes con poco a ningún conocimiento previo de programación, estudiantes de escuelas secundarias, universidades, escuelas vocacionales;
- profesionales de la industria que deseen explorar tecnologías que estén conectadas con Python o que utilicen Python como base para proyectos más complejos;
- líderes de equipo, gerentes de productos y gerentes de proyectos que desean comprender la terminología y los procesos en el ciclo de desarrollo de software para administrar y comunicarse de manera más efectiva con los equipos de TI, pruebas y desarrollo;
- simplemente cualquier persona interesada en aprender a programar por diversión o con fines relacionados con el trabajo.

Prerrequisitos

No hay prerrequisitos para este curso. El único previo es la habilidad para usar una computadora personal y conocimientos muy básicos en matemáticas.

¡Todos son bienvenidos a registrarse!

Certificación Alineada

El plan de estudios de Python Essentials 1 ayuda a los estudiantes a prepararse para PCEP – Certified Entry-Level Python Programmer. La certificación **PCEP (Exam PCEP-30-0x)** es una credencial profesional que mide la capacidad del candidato para realizar tareas de codificación relacionadas con los elementos esenciales de la programación en el lenguaje Python. Un candidato de prueba debe demostrar un conocimiento suficiente de los conceptos universales de la programación informática, la sintaxis y la semántica del lenguaje Python, así como las habilidades para resolver los desafíos típicos de implementación con la ayuda de la biblioteca estándar de Python.

Descripción del curso

Python Essentials 1 cubre todos los aspectos básicos de la programación en Python, así como conceptos y técnicas generales de programación de computadoras. El curso familiariza al estudiante con el enfoque de la programación y cubre los siguientes temas:

- Términos fundamentales, conceptos y definiciones.
- Sintaxis básica de Python, semántica y entorno de tiempos de ejecución;
- Literales, variables, sistemas numéricos, operadores y tipos de datos;
- Flujo de control: ejecución condicional y bucles;
- Colecciones de datos: listas, tuplas, diccionarios y cadenas;
- Funciones;
- Excepciones, solución de problemas y depuración.

El curso se divide en **4 módulos**. Cada estudiante tiene acceso a materiales prácticos, cuestionarios y pruebas para aprender a utilizar las habilidades de los conocimientos adquiridos en el curso e interactuar con algunas tareas y situaciones de programación de la vida real.

Objetivos del curso

1. Desarrollar una conciencia de los lenguajes de programación.
   - ✓ 1.1 Identificar enfoques de diseño de lenguajes de programación.
   - ✓ 1.2 Explicar los componentes de los lenguajes de programación.
   - ✓ 1.3 Examinar las conexiones entre los elementos de las matemáticas y la informática, incluidos los números binarios, la lógica (incluida el algebra booleana), los conjuntos y las funciones.

2. Demostrar competencia en el uso de programas informáticos especializados.
   - ✓ 2.1 Usar software de programación de computadora especializado para resolver problemas (por ejemplo: editor, consola, IDLE, IDE, depurador)
   - ✓ 2.2 Demostrar competencia en el uso de software informático especializado (por ejemplo: Python, entorno de programación interactivo **Edube**)

3. Demostrar conocimiento, habilidades y aplicación de TI y sistemas de comunicación para lograr los objetivos laborales y mejorar el desempeño en el lugar de trabajo.
   - ✓ 3.1 Desarrollar habilidades de teclado para ingresar y manipular texto y datos (por ejemplo: usar atajos de teclado, escribir código)
   - ✓ 3.2 Describir y utilizar software y tecnología informática actuales y emergentes para realizar tareas personales y comerciales (por ejemplo: preparar una lista de tareas pendientes, planificar un proyecto, escribir pseudocódigo)
   - ✓ 3.3 Realizar una variedad de operaciones, como clasificar y filtrar datos, organizar y mostrar información en una variedad de formas, como formatos de números (conversión a diferentes sistemas numéricos o unidades)

4. Demostrar competencia en el uso de aplicaciones de software comunes.
   - ✓ 4.1 Comparar y contrastar el uso apropiado de varias aplicaciones y programas de software (por ejemplo: compilador versus intérprete).
   - ✓ 4.2 Demostrar competencia en el uso de diversas aplicaciones y programas de software (por ejemplo: editores de código)
   - ✓ 4.3 Explicar por qué existen diferentes tipos de archivos (por ejemplo: archivos fuente, archivos semi-compilados, archivos **.py**)
   - ✓ 4.4 Identificar los tipos de contenido asociado con diferentes tipos de archivos.

5. Demostrar comprensión y comunicación.
   - ✓ 5.1 Usar habilidades y estrategias de lectura y escritura para comunicarse de manera efectiva (leer documentación, escribir código con comentarios)
   - ✓ 5.2 Organizar ideas y comunicar mensajes escritos (por ejemplo: usando comentarios)
   - ✓ 5.3 Reconocer que más de un algoritmo puede resolver un problema dado.
   - ✓ 5.4 Crear un programa que implemente un algoritmo para lograr un objetivo determinado.

6. Explorar las características, tareas, trayectorias profesionales y herramientas disponibles para iniciar una carrera en desarrollo de software y tecnologías relacionadas.

- ✓ 6.1 Explorar una variedad de carreras que utilizan habilidades informáticas y algorítmicas, e investigar oportunidades de carrera en programación y tecnologías relacionadas.
- ✓ 6.2 Analizar el impacto de la informática en los negocios, el comercio y las organizaciones (por ejemplo: procesamiento de inventario automatizado, transacciones financieras, virtualización, computación en la nube).
- ✓ 6.3 Discutir las responsabilidades éticas del programador y evaluar los impactos del plagio y la infracción de derechos de autor en sus proyectos, vida y carrera.
- ✓ 6.4 Identificar y analizar tareas realizadas por programadores y testers.
- ✓ 6.5 Explicar la necesidad de formación continua de los programadores.
- ✓ 6.6 Comprender, identificar y utilizar las herramientas y métodos adecuados para apoyar el aprendizaje permanente.
- ✓ 6.7 Identificar las credenciales y certificaciones que pueden mejorar la empleabilidad de un programador de computadoras.

7. Resolver problemas utilizando habilidades de pensamiento crítico, creatividad e innovación.

- ✓ 7.1 Emplear habilidades de pensamiento crítico para resolver problemas y tomar decisiones.
- ✓ 7.2 Realizar investigaciones técnicas para recabar la información necesaria para la toma de decisiones.
- ✓ 7.3 Discutir herramientas o recursos digitales para usar en una tarea del mundo real en función de su eficiencia y eficacia.

8. Escriba y documente el código utilizando las pautas, las convenciones de codificación y las mejores prácticas de **PEP8**.

- ✓ 8.1 Usar convenciones de nomenclatura apropiadas para definir variables de programa, método, funciones y clases.
- ✓ 8.2 Usar documentación interna (por ejemplo: comentarios de una o varias líneas, cadenas de documentación del módulo) para documentar un programa de acuerdo en los estándares aceptados.
- ✓ 8.3 Enumerar ejemplos de buenas prácticas de programación en Python (por ejemplo: escribir código auto comentado).
- ✓ 8.4 Utilizar el formato y diseño de código adecuados (por ejemplo: sin exceder la longitud de línea recomendada).
- ✓ 8.5 Comprender la importancia de usar convenciones de codificación; use los lineamientos más importantes de **PEP8**.

9. Demostrar competencia en programación básica y Python.

- ✓ 9.1 Describir la estructura de un programa simple, explique por qué es importante la secuencia.
- ✓ 9.2 Definir el término ***algoritmo*** y explique cómo se relaciona con la resolución de problemas.
- ✓ 9.3 Describir, crear y utilizar estructuras de programación iterativas (por ejemplo: bucles)
- ✓ 9.4 Describir, crear y usar estructuras condicionales y explicar la lógica utilizada para ellas (if, else, else-if, finally)
- ✓ 9.5 Explicar los tipos y el uso de variables en la programación (por ejemplo: nombrar, crear, inicializar y modificar variables en Python).
- ✓ 9.6 Escribir programas sencillos en pseudocódigos.
- ✓ 9.7 Buscar, analizar, describir, solucionar y depurar errores en el código.

- ✓ 9.8 Comprender los conceptos fundamentales de programación tales como: compilador, intérprete, código fuente, código máquina, IDE, etc. Utilizar habilidades para instalar y configurar herramientas básicas de desarrollo.
- ✓ 9.9 Comprender la diferencia entre la sintaxis y la semántica de un lenguaje de programación; describir los tipos de datos básicos, identificadores y palabras clave en Python.
- ✓ 9.10 Comprender la historia de desarrollo de Python, sus principales rasgos y características.
- ✓ 9.11 Diseñar, crear, editar y ejecutar archivos fuente de Python usando IDLE.
- ✓ 9.12 Comprender y utilizar los literales numéricos de Python, su sintaxis, tipos y formatos.
- ✓ 9.13 Comprender y utilizar operadores de asignación, operadores aritméticos y expresiones de Python.
- ✓ 9.14 Realizar operaciones básicas de I/O en programas de Python.
- ✓ 9.15 Comprender y utilizar los tipos de datos booleanos y sus características.
- ✓ 9.16 Comprender y utilizar operadores relacionados en Python.
- ✓ 9.17 Realizar operaciones bit a bit en Python.
- ✓ 9.18 Comprender el rol de las listas y operar con ellas para realizar indexaciones, cortes y manipulación de contenido.
- ✓ 9.19 Comprender y utilizar listas multidimensionales en Python.
- ✓ 9.20 Comprender y aplicar el concepto de funciones, y ser capaz de crear e invocar funciones creadas por el usuario.
- ✓ 9.21 Comprender y aplicar las principales características de la programación estructural.
- ✓ 9.22 Comprender el concepto de alcances de nombres y ser capaz de distinguir variables globales y locales, así como comprender cómo funciona el sombreado de nombres.
- ✓ 9.23 Comprender y utilizar los principios de las tuplas, incluida la noción de inmutabilidad.
- ✓ 9.24 Comprender el papel de los diccionarios, y ser capaz de utilizarlos de forma eficaz en las circunstancias adecuadas.
- ✓ 9.25 Comprender el papel de las cadenas, y saber manipularlas utilizando técnicas básicas de procesamiento.
- ✓ 9.26 Comprender la función de los módulos en Python y conocer las formas disponibles de importar módulos al código/espacio de nombres.
- ✓ 9.27 Usar módulos Python estándar útiles seleccionados.
- ✓ 9.28 Comprender y utilizar las formas de identificar y manejar errores de tiempo de ejecución en Python.
- ✓ 9.29 Comprender el propósito de las sentencias de control para el manejo de excepciones (try, except).
- ✓ 9.30 Usar Python en aplicaciones de la vida cotidiana, incluidas las actividades DIY.
- ✓ 9.31 Usar Python en varios escenarios relacionados con el trabajo.

10 Realizar actividades de diseño, codificación y prueba de programas.

- ✓ 10.1 Demostrar conocimiento de conceptos y terminología informática.
- ✓ 10.2 Crear y documentar un diseño de programa.
- ✓ 10.3 Comunicar las especificaciones de diseño.
- ✓ 10.4 Desarrollar prototipo.
- ✓ 10.5 Realizar revisiones y mejoras de los sistemas de software (por ejemplo: refactorización de código, actualizaciones, add-ons)
- ✓ 10.6 Demostrar uso competente de herramientas de desarrollo de programación.
- ✓ 10.7 Desarrollar código de pruebas validando entradas, resultados esperados, determinando casos de prueba límite.
- ✓ 10.8 Revisar el código del programa y hacer arreglos necesarios.

11. Usar habilidades de comunicación oral y escrita para crear, expresar e interpretar información e ideas.

- ✓ 11.1 Seleccionar y emplear conceptos y estrategias de comunicación apropiados para mejorar la comunicación oral y escrita en el lugar de trabajo.
- ✓ 11.2 Localizar, organizar, analizar, adoptar, adaptar y hacer referencia a información escrita de varias fuentes (por ejemplo: documentación)
- ✓ 11.3 Construir la comunicación utilizando la terminología de desarrollo adecuada.
- ✓ 11.4 Analizar los impactos positivos (por ejemplo: la terminología sensible como amo-esclavo, lista negra-lista blanca; interconexión de cosas a través de la tecnología, etc).
- ✓ 11.5 Discutir la influencia de la tecnología en la forma en que las personas construyen, desarrollan y administran organizaciones y proyectos.

12. Crear y documentar un programa de computadora que use una variedad de estructuras internas y de control para manipular varios tipos de datos.

- ✓ 12.1 Usar un editor de programas para escribir el código fuente de un programa.
- ✓ 12.2 Escribir programas que utilicen estructuras anidadas (por ejemplo: recursiones, bucles anidados, sentencias if anidadas).
- ✓ 12.3 Documentar un programa y usar un estilo consistente para aumentar la legibilidad, mejorar el mantenimiento y explicar el propósito y la operación de los elementos del programa.
- ✓ 12.4 Escribir programas que usen una variedad de tipos de datos y operadores comunes.
- ✓ 12.5 Escribir programas que realicen conversión de datos entre tipos de datos numéricos y de cadena, y otros tipos de datos (por ejemplo: encasillamiento).
- ✓ 12.6 Escribir programas y funciones que acepten y procesen argumentos.
- ✓ 12.7 Escribir programas y funciones que devuelvan valores.
- ✓ 12.8 Participar en la revisión de código pro pares para verificar la funcionalidad del programa, los estilos de programación, la facilidad de uso del programa y el cumplimiento de los estándares de programación comunes.

Requisitos para el equipo

Se puede acceder al curso en línea a través de cualquier navegador de internet. El único requisito previo es una conexión en línea activa, una computadora portátil/computadora de escritorio/tableta/dispositivo móvil y un navegador de internet. Pata obtener la mayor experiencia de aprendizaje, recomendamos utilizar las versiones más recientes de **Mozilla Firefox, Internet Explorer / Microsoft Edge o Google Chrome**.

Habilidades obtenidas en el curso

Un participante del curso que complete **Python Essentials 1** podrá adquirir las siguientes habilidades:

- ✓ Capacidad para diseñar, desarrollar y mejorar programas informáticos sencillos escritos en Python;
- ✓ Un conocimiento adecuado para comenzar a aprender otro lenguaje de programación;

- ✓ Competencia suficiente para el examen PCEPTM y obtener la certificación correspondiente;
- ✓ Las habilidades necesarias para participar en el curso **Python Essentials 2**;
- ✓ Habilidades y conocimientos que le permitan aceptar un trabajo en un nivel de entrada;
- ✓ La posibilidad de continuar su desarrollo personal a través de la autoeducación y la superación personal.

La siguiente tabla detalla los módulos del curso y sus competencias asociadas. Cada módulo es una unidad integrada de aprendizaje que consta de contenido, actividades y evaluaciones que apuntan a un conjunto específico de competencias. El tamaño del módulo depende de la profundidad del conocimiento.

Módulo 1: Introducción a Python y a la programación de computadoras
Después de completar el Módulo 1, el estudiante: • tendrá un conocimiento básico de programación informática y desarrollo de software; • comprenderá los conceptos fundamentales de programación tales como: compilador, intérprete, código fuente, código máquina, IDE; • tendrá una orientación en la historia del desarrollo de Python, sus principales rasgos y sus características; • adquirirá habilidades que le permitirán instalar y configurar herramientas básicas de desarrollo, así como códigos y ejecutar el primer programa Python.
Módulo 2: Tipos de datos, variables, operadores y operaciones básicas de E/S de Python.
Después de completar el Módulo 2, el estudiante: • adquirirá habilidades que le permitirán crear, editar y ejecutar archivos fuente de Python usando IDLE; • tendrá algún conocimiento de los literales numéricos de Python, su sintaxis, tipos y formatos; • tendrá orientación en temas relacionados con operadores y expresiones aritméticas de Python; • adquirirá la capacidad de nombrar, crear, inicializar y modificar variables; • tendrá habilidades que le permitirán realizar operaciones básicas de entrada/salida en un programa de Python.
Módulo 3: Valores booleanos, ejecución condicional, bucles, listas y procesamientos de listas. Operaciones lógicas y bit a bit.
Después de completar el **Módulo 3**, el estudiante: • conocerá las características básicas del tipo de datos booleano; • adquirirá habilidades para utilizar con operadores relacionados en Python; • tendrá la capacidad de utilizar eficazmente las sentencias de control: **if, if-else, ifelif-else;** • comprenderá el papel de un bucle y será capaz de utilizar las sentencias de control: **while, for;** • tendrá una orientación en operaciones **bit a bit en Python;** • conocerá la función de las listas y será capaz de operar con ellas para realizar acciones que incluyen indexación, división y manipulación de contenido; • tendrá conocimientos de listas multidimensionales en Python;

Módulo 4: Funciones, Tuplas, Diccionarios, Excepciones y Procesamiento de datos.

Después de completar el **Módulo 4**, el estudiante:

- entenderá el concepto de funciones y será capaz de codificar y llamar a sus propias funciones.
- tendrá una orientación de las principales características de la programación estructural;
- tendrá un conocimiento esencial de los alcances de los nombres y será capaz de distinguir las variables globales y locales, así como entender cómo funciona el sombreado de nombres;
- comprenderá los principios de las tuplas, incluida la noción de inmutabilidad;
- conocerá el papel de los diccionarios y será capaz de usarlos de manera efectiva en las circunstancias apropiadas;
- entenderá el mecanismo de manejo de excepciones en Python;
- sabrá cómo usar las excepciones integradas de Python más importantes;
- creará e implementará excepciones definidas por el usuario para asegurar un flujo estable de un programa de computadora.

Examen Final

La prueba final se basa en lo que el participante ha aprendido en todo el curso, después de completar esta parte, el estudiante:

- revisará la información más importante que ha leído y probará las habilidades y conocimientos que ha adquirido a lo largo del curso;
- comprenderá mayor el nivel de competencia necesaria para aprobar el examen de certificación **PCEP** y obtendrá la certificación correspondiente.