Ciclo 1-Fundamentos de Programación Grupo 66

Descripción

Duración:

Duración: 200 horas (7 semanas de tiempo completo).

- 50 horas de sesiones sincronizas de programación guiadas por un profesor.
- 120 horas de trabajo individual de programación, con apoyo opcional de tutores y herramientas virtuales.
- 25 horas de formación de lectura en inglés.
- 5 horas de trabajo en habilidades personales (coaching) guiadas por un profesor.

Perfil del egresado:

El estudiante que haya culminado con éxito este ciclo estará en la capacidad de:

- Desarrollar un programa monousuario para resolver los requerimientos planteados por un tercero.
- Construir un programa trabajando de manera individual.
- Construir un programa utilizando un lenguaje imperativo (Python).
- Construir un programa siguiendo el ciclo completo de vida de desarrollo, que comienza con la identificación y documentación de los requerimientos funcionales y termina con un conjunto de pruebas unitarias.
- Construir un programa con una interfaz de consola o gráfica simple.
- Construir un programa que maneje estructuras de datos lineales en memoria principal.
- Construir un programa que maneje archivos de texto para almacenar información persistente (formato JSON / CSV).

Evaluación, retroalimentación y plagio:

La evaluación del ciclo 1 y la retroalimentación del estudiante se explicará en los siguientes puntos:

- La evaluación del ciclo 1 corresponde a la solución y aprobación de 5 retos que serán publicados al final de cada semana junto con la evaluación del curso de inglés.

- El estudiante debe subir a la plataforma de evaluación la solución del reto correspondiente. La plataforma ejecutará las pruebas unitarias y dará una calificación. Esta calificación corresponde al número de requerimientos perfectamente resueltos (que pasen todas las pruebas). La plataforma le señalara al estudiante las pruebas que no se cumplieron satisfactoriamente.
- La plataforma revisará las soluciones entregadas por los estudiantes con el propósito de detectar semejanzas que puedan ser interpretadas como copia. En ese caso dichas soluciones serán revisadas individualmente.
- La nota final del ciclo se calcula de la siguiente manera: reto 1 (10%), reto 2 (10%), reto 3 (20%), reto 4 (20%), reto 5 (20%), evaluación de inglés (20%).
- La nota mínima aprobatoria para este ciclo es 3 sobre 5.

Metodología:

El ciclo 1 – Fundamentos de programación consta de una clase diaria acorde al horario entregado en cada grupo. El lenguaje de programación utilizado es Python. El curso está organizado en 7 semanas en las cuales el profesor trabajará durante la clase utilizando ejemplos que muestran los conceptos de la semana y en las sesiones asincrónicas se dará material de refuerzo a lo visto en la sesión sincrónica. A continuación, se presentarán los conceptos a desarrollar en cada una de las semanas:

Unidad 1.

Hola mundo - primeros pasos para programar

1. Conocer el entorno de desarrollo y los ambientes de trabajo en Python (Spider – Jupyter Notebook).
2. Algoritmos: Conceptualización, Representación y Codificación de soluciones).
3. Creación de variables utilizando e identificados los diferentes tipos de datos en Python junto con el uso de operaciones básicas del lenguaje.
4. Aprender a llamar/invocar funciones básicas de Python, comprendiendo la lógica y sintaxis.
5. Desarrollar funciones propias en Python, utilizando conceptos tales como variables locales y parámetros.
6. Llamar funciones con parámetros, y funciones desde otras funciones.
7. Creación y utilización de módulos para la agrupar funciones relacionadas.

Unidad 2.

Condicionales y estructuras propias de Python (Diccionarios)

1. Conocer el uso de booleanos y sus operadores.
2. Realizar expresiones relacionales y lógicas (álgebra booleana).
3. Creación de instrucciones con condicionales.
4. Tipificación de condicionales.
5. Cadenas de caracteres (strings) y operaciones sobre estas.
6. Encapsulando y pasando información entre funciones a través de diccionarios.
7. Operaciones básicas con diccionarios.
8. Mutabilidad, borrado de datos y parámetros por referencia.

8. Manipulación de entrada y salida de información en un programa.

Unidad 3.
Estructuras de datos y ciclos

1. Comprender y utilizar las diferentes instrucciones iterativas para la solución de problemas en sus programas (while, for, y do while).
2. Entender el concepto de indexación en strings.
3. Crear, manipular y utilizar listas en Python como una nueva estructura de datos.
4. Utilizar la anidación de listas para representar relaciones, y composición de colecciones: listas paralelas y listas compuestas.
5. Comprender y utilizar tuplas, conjunto y diccionarios, para agrupar elementos en múltiples contextos, apreciando las diferencias de estas estructuras de datos.
6. Aprender y utilizar patrones de recorrido sobre estructuras como cadenas, tuplas, listas, conjuntos y diccionarios.
7. Construir una aplicación CRUD básica con interfaz de consola,

combinando los conocimientos adquiridos referentes a las colecciones de datos nativas de Python

Unidad 4.
Manipulación de Colecciones y Librerías

1. Convirtiendo colecciones de datos (cadenas en listas, listas en cadenas, listas en tuplas, tuplas en listas, etc.).
2. Características de Python para aprovechar el paradigma de programación funcional.
3. Funciones de primera clase, funciones sin nombre y funciones de orden superior.
4. Generalizando la manipulación de colecciones con las funciones map, filter, zip y reduce
5. Decisiones generalizadas sobre colecciones: funciones any y all.
6. Librería Numpy para el uso de matrices como nueva estructura de datos de 2 dimensiones.
7. Pandas para manipular conjunto de datos.

Unidad 5.
Visualización y manipulación de archivos

1. Aprender a manejar archivos, es decir, cómo leer y escribir archivos, para utilizarlos desde los programas (archivos JSON, CSV, Excel).
2. Agregar perdurabilidad de los datos a una aplicación CRUD con interfaz de consola, explotando la relación entre diccionarios y JSON.
3. Comprender la utilidad de

Unidad 6.
Programación orientada a objetos en Python e interfaces gráficas

1. Familiarización con el Paradigma de Programación Orientada a Objetos en Python.
2. Diferenciación entre interfaz y mundo del problema modelando con objetos en Python.
3. Relacionado objetos e interfaces gráficas con la librería Turtle.
4. Construcción de una aplicación sencilla con interfaz y objetos

herramientas de visualización de datos y procesamiento numérico.

4. Hacer uso de matplotlyb para la visualización de datos.

empleando EasyGUI o TKINTER de Python.

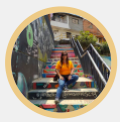
Avisos

Avisos

Unidad 1 ▶

Profesor

Participantes



Ver todos

Escucha esta página utilizando ReadSpeaker