Tecnicatura Universitaria en Programación

Programación I

Planificación 2º Cuatrimestre – Ciclo Académico 2025

1- DATOS ADMINISTRATIVOS DE LA ASIGNATURA

Carrera:	Tecnicatura Superior en Programación
Asignatura:	Programación I
Nivel de la carrera:	1º cuatrimestre

Carga horaria presencial semanal:	8	**% Horas presenciales:**	-
Carga horaria no presencial semanal:	0	**% Horas no presenciales:**	-
Carga horaria total:	8	**% Horas total:**	-

Profesoras/es	
Apellido(s) y nombre(s)	**Cargo docente**
Arrollo, Carlos	Prof. Ayudante
Bartoncello, Ricardo	Prof. Ayudante
Carbonari, Verónica	Prof. Ayudante
Cardozo, Natali	Prof. Ayudante
Costello, Patricio	Prof Ayudante
Di Leone, Agustín	Prof. Ayudante
Falcone, Facundo	Profesor
Fernandez, David	Profesor
Fernández, Luis	Profesor
Fernandez, Mariano	Profesor
Ferrini, Lucas	Prof. Ayudante
Gatto, Catriel	Prof. Ayudante
Gonzalez, Gisele	Prof. Ayudante
Guevara, Mariano	Prof. Ayudante
Gutiérrez, Mariel	Prof. Ayudante
Mansilla, Yhamil	Prof. Ayudante
Meloni, Albana	Prof. Ayudante
Morán, Ezequiel	Prof. Ayudante
Ochoa, Gonzalo	Prof. Ayudante
Panello, Fausto	Prof. Ayudante

Pavlov, Valeriy	Profesor
Samaniego, Manuel	Prof. Ayudante
Scarafilo, Germán	Profesor
Scudero, Yanina	Profesor
Tulis, Luis	Profesor
Zotti, Enzo	Profesor

2- FUNDAMENTACIÓN Y PRESENTACIÓN DE LA ASIGNATURA

La asignatura tiene como objetivo proporcionar a los estudiantes los conocimientos y habilidades necesarias para desarrollar programas utilizando el lenguaje de programación Python

La asignatura se enfoca en el aprendizaje a través de la práctica y el desarrollo de proyectos, con el fin de que los estudiantes puedan aplicar los conocimientos teóricos adquiridos en la resolución de problemas reales. Además, se enfatiza en el desarrollo de habilidades de resolución de problemas y pensamiento crítico.

Se tiene como objetivo familiarizar a los estudiantes con las herramientas y técnicas necesarias para el desarrollo de aplicaciones utilizando Python, incluyendo la instalación y configuración de un ambiente de desarrollo, el uso de bibliotecas y frameworks, y la creación de aplicaciones.

3- OBJETIVOS DE APRENDIZAJE

- Aprender a programar en Python y conocer sus principales características y aplicaciones.
- Conocer y aplicar los conceptos fundamentales de la programación: variables, tipos de datos, estructuras de control, estructuras algorítmicas y funciones.
- Desarrollar habilidades en la creación de algoritmos y la resolución de problemas utilizando programación en Python.
- Desarrollar un proyecto práctico utilizando las herramientas y técnicas aprendidas en el curso.
- Fomentar el trabajo en equipo y la colaboración en la resolución de problemas y proyectos prácticos.

4- PROGRAMA ANALÍTICO

I. Fundamentos de Programación:

A. Introducción a los paradigmas de programación (Estructurado, Orientado a objetos y Funcional)
B. Lenguajes compilados e interpretados.
C. Lenguaje de programación Python.
D. Variables y tipos de datos en Python.
E. Operadores y expresiones.
F. Estructuras condicionales.
G. Estructuras repetitivas.
H. Reglas de estilo.
I. Control de versiones y repositorios: Git y Github.

II. Funciones:

A. Definición de función.
B. Beneficios en la implementación de funciones.
C. Parametrización y retorno.
D. Variables globales y locales.
E. Mutabilidad e inmutabilidad de las variables.
F. Reserva de memoria.
G. Documentación.
H. Módulos y paquetes.
I. Funciones recursivas. Ventajas y desventajas.

III. Tipos de datos avanzados (Parte 1):

A. Arreglos unidimensionales: declaración, acceso, carga de elementos y búsquedas.
B. Arreglos bidimensionales: declaración, acceso, carga de elementos y búsquedas.
C. Algoritmos de ordenamiento.

IV. Cadenas de caracteres:

A. Manipulación de cadenas de caracteres desde el punto de vista algorítmico.
B. Funciones propias para el manejo de caracteres.

V. Tipos de datos avanzados (Parte 2):

A. Listas. Métodos de manipulación de listas en Python.
B. Sets y tuplas
C. Datos estructurados: diccionarios en Python.

VI. Archivos:

A. Lectura, escritura y manipulación de archivos de texto.
B. Lectura, escritura y manipulación de archivos csv.
C. Lectura, escritura y manipulación de archivos JSON.

VII. Paradigma funcional:

A. Ciudadanos de primera clase.
B. Pasaje de funciones como parámetros de otras funciones.

VIII. Estructura de un juego en consola:

A. Introducción al concepto de video juego.
B. Prácticas para la resolución de juegos en consola.

IX. Pygame:

A. Instalación del módulo.
B. Ciclo de vida de un juego con Pygame.
C. Carga y escalado de imágenes.
D. Posicionamiento (coordenadas).
E. Figuras y superficies.
F. Eventos del teclado y mouse.
G. Colisiones.
H. Temporizadores.
I. Sonidos.

5- METODOLOGÍA DE ENSEÑANZA

- **Presentación del temario:** se presentará el temario del curso de manera detallada para que los estudiantes tengan una idea clara de los temas que se van a tratar.
- **Clases teóricas:** se impartirán clases teóricas donde se explicarán los conceptos y las técnicas necesarias para el desarrollo de aplicaciones.
- **Clases prácticas:** se realizarán ejercicios y proyectos prácticos para que los estudiantes puedan aplicar los conocimientos adquiridos y desarrollar habilidades de programación.
- **Tutorías y apoyo:** se ofrecerá apoyo y tutorías para los estudiantes que necesiten ayuda adicional en la comprensión de los temas o en la realización de los proyectos.
- **Evaluaciones:** se realizarán evaluaciones periódicas para medir el progreso de los estudiantes y asegurarse de que estén comprendiendo los temas y desarrollando habilidades de programación adecuadas.
- **Trabajo práctico integrador:** se propondrá un proyecto final donde los estudiantes puedan aplicar todos los conocimientos y habilidades adquiridas en el curso para desarrollar un videojuego utilizando Python y Pygame.

6- METODOLOGÍA DE EVALUACIÓN

- **Evaluaciones formativas:** se trata de evaluaciones periódicas a lo largo del curso, con el objetivo de brindar retroalimentación al estudiante sobre su progreso y desempeño. Estas evaluaciones pueden ser exámenes, tareas, proyectos, entre otros.
- **Evaluaciones sumativas:** se trata de evaluaciones que se realizan al final de un periodo determinado, con el objetivo de medir el nivel de conocimiento y habilidades adquiridos por el estudiante. Estas evaluaciones pueden ser exámenes finales, presentaciones de proyectos, trabajos escritos, entre otros.
- **Autoevaluación:** se trata de una evaluación en la que el estudiante reflexiona sobre su propio proceso de aprendizaje y desempeño en el curso, identificando fortalezas y debilidades y estableciendo estrategias para mejorar.

- **Coevaluación**: se trata de una evaluación en la que los estudiantes se evalúan mutuamente, identificando fortalezas y debilidades de cada uno y brindando retroalimentación constructiva.

7- CRONOGRAMA SINTÉTICO (TENTATIVO)

Semana	Unidad/Bloque temático
Semana 1	Fundamentos de programación.
Semana 2	Estructura repetitiva While.
Semana 3	Estructura repetitiva For.
	Funciones.
Semana 4	Funciones recursivas.
	Control de versiones.
Semana 5	Tipos de datos avanzados (arreglos unidimensionales)
Semana 6	Tipos de datos avanzados (arreglos bidimensionales)
Semana 7	Algoritmos de ordenamiento.
Semana 8	Cadenas de caracteres.
Semana 9	Primer Parcial.
	Tipos de datos avanzados (listas).
Semana 10	Tipos de datos avanzados (tuplas, sets y diccionarios).
	Recuperatorio Primer Parcial.
Semana 11	Archivos
	Paradigma funcional
Semana 12	Estructura de un juego en consola
Semana 13	Biblioteca Pygame
Semana 14	Biblioteca Pygame
Semana 15	Biblioteca Pygame
Semana 16	Segundo Parcial.
	Recuperatorio Segundo Parcial.

8- RECURSOS NECESARIOS

- Una PC o laptop con acceso a Internet para acceder al material del curso, ver videos y resolver ejercicios.Tipos de datos avanzados: Arreglos unidimensionales

- Un entorno de programación instalado en el equipo de trabajo, como PyCharm o Visual Studio Code, según el lenguaje de programación que se utilice.
- Libros, artículos y tutoriales en línea para complementar el material del curso y profundizar en temas específicos.
- Bibliotecas de software, como por ejemplo Pygame, etc., según los temas del curso.
- Foros y comunidades en línea donde los estudiantes puedan hacer preguntas y compartir sus proyectos.
- Un sistema de gestión de versiones como Git, para ayudar a los estudiantes a mantener un registro de su código y colaborar en proyectos en grupo.

9- REFERENCIAS BIBLIOGRÁFICAS

- El libro de Python: "Python Cookbook, Third edition" de David Beazley y Brian K. Jones.
- "Pygame Documentation" (https://www.pygame.org/docs/)
- Material de la cátedra.