Facultad de Informática | UNIVERSIDAD NACIONAL DE LA PLATA

Taller de lenguajes

Carrera/Plan:

Ciencia de Datos en Organizaciones Plan 2024

Año 2026

Año: 2
Régimen de Cursada: Semestral
Carácter (Obligatoria/Optativa):
Correlativas: Algoritmos y Programación II
Profesor/es: Sofía Martin
Hs. semanales: 6h

FUNDAMENTACIÓN

Tiene como propósito consolidar y aplicar los conceptos fundamentales sobre programación adquiridos durante el primer año de la carrera, con un énfasis particular en el trabajo práctico sobre la computadora. Se busca que los estudiantes refuercen estos conocimientos mediante actividades que integren la teoría con su implementación en un lenguaje de programación, poniendo énfasis en el análisis formal de las características del lenguaje y su comparación con los que el estudiante conociera. Este enfoque permitirá ampliar su perspectiva sobre las herramientas y paradigmas de programación, favoreciendo su adaptabilidad y fortaleciendo sus habilidades técnicas para enfrentar nuevos desafíos.

OBJETIVOS GENERALES:

- Facilitar la creación de una aplicación concreta para profundizar los conocimientos adquiridos en los cursos iniciales de Algoritmos y Programación.
- Promover un abordaje teórico-práctico del lenguaje de programación Python, destacando el análisis formal de sus características.
- Fomentar la comparación de Python con los lenguajes previamente estudiados para comprender sus diferencias y similitudes.
- Desarrollar habilidades en el uso de reglas de estilo y buenas prácticas de codificación para escribir código limpio y eficiente.
- Incentivar el respeto y la aplicación de estándares definidos en el desarrollo de software.
- Introducir y fomentar el uso de herramientas de versionado de código.
- Enseñar y reforzar las prácticas adecuadas para documentar correctamente un desarrollo.
- Promover habilidades de trabajo en equipo, necesarias para proyectos colaborativos.
- Desarrollar competencias para presentar y exponer el trabajo realizado de manera clara y profesional.

CONTENIDOS MINIMOS (de acuerdo al Plan de Estudios)

Estudio de un lenguaje de programación en el que se desarrollen aplicaciones concretas.

PROGRAMA ANALÍTICO

Unidad I
Conceptos de software y recursos libres. El proceso de ejecución de un programa escrito en Python. Entornos virtuales y otras herramientas complementarias para el desarrollo de software.

Unidad II
Sintaxis básica del lenguaje Python. Tipos predefinidos. Estructuras de control. La estructura de un programa de Python. Definición de funciones y módulos. Pasaje de parámetros.

Unidad III
Estructuras de datos básicas: listas, tuplas, conjuntos y diccionarios.

Unidad IV
Manejo de archivos. Procesamiento y análisis de datos en diferentes formatos como JSON y CSV.

Unidad V
Análisis y uso de librerías externas para la creación de aplicaciones web orientadas al análisis de datos.

Unidad VI
Conceptos básicos de programación orientada a objetos y manejo de excepciones.

Unidad VII
Programación de análisis y vizualizaciones interactivas. Características generales. Manejo de eventos.

Unidad VIII
Introducción al análisis y visualización de datos.

Unidad IX.
Aspectos básicos de accesibilidad web: pautas e iniciativas; recomendaciones. Validadores.

BIBLIOGRAFÍA

- Python Programming: An Introduction to Computer Science. John M. Zelle
- Introduction to Computing and Programming in Python, A Multimedia Approach. Mark Guzdial.
- Learning Python Application Development. Ninad Sathaye
- Beginning Python: From Novice to Professional - Magnus Lie Hetland.
- An Introduction to Python. Guido van Rossum.
- Learning Python. O'Reilly.
- Tutorial de Python en castellano: https://docs.python.org/es/3/tutorial/index.html
- Automate the boring stuff with python - practical programming for total beginners. AL SWEIGART

- Introducing Data Science. Big Data , Machine Learning, and more, using Python tools. DAVY CIELEN, ARNO D. B. MEYSMAN, MOHAMED ALI. Manning.
- Principles of Data Science. Sinan Ozdemir. Packt Publishing Ltd.
- Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. McKinney, W. O'Reilly Media.

METODOLOGÍA DE ENSEÑANZA

Al inicio de la cursada, se implementará una evaluación diagnóstica mediante una encuesta en línea. Este instrumento permitirá identificar los conocimientos previos de los estudiantes, así como recopilar información adicional sobre su contexto, como su situación laboral y condiciones de conectividad.

La propuesta adopta una modalidad de taller, donde teoría y práctica se integran de manera dinámica y complementaria. Se retomarán los conceptos aprendidos en las asignaturas previas de programación, profundizándolos a través de su aplicación práctica en el lenguaje Python. Este enfoque comparativo busca no solo reforzar los aprendizajes previos, sino también adaptarlos y contextualizarlos en función de las características particulares de este lenguaje.

Las clases teóricas se estructuran como encuentros semanales en los que se desarrollan los conceptos centrales de cada tema, priorizando el análisis crítico y la adquisición de habilidades para aplicar técnicas y herramientas informáticas. Se busca que estas herramientas estén alineadas, en la medida de lo posible, con las tendencias actuales impulsadas por el cambio tecnológico.
Para acompañar este proceso, se proporcionan materiales que incluyen casos prácticos, permitiendo a los estudiantes analizar, seleccionar y emplear de manera eficiente las técnicas y herramientas más adecuadas para cada problema planteado.
Además, se proponen actividades específicas orientadas a:

- Examinar tecnologías existentes,
- Evaluar posibles soluciones,
- Explorar la evolución de las soluciones frente a un mismo problema,
- Comparar la eficiencia entre distintas alternativas,
- Identificar y evidenciar errores potenciales en los procesos de resolución.

Estas instancias buscan fomentar un aprendizaje activo, desafiante y vinculado a escenarios reales, promoviendo tanto la reflexión como la capacidad crítica de los estudiantes.

Durante los horarios de práctica, se desarrollan actividades planificadas en las que los estudiantes enfrentan desafíos que deben transformar en *ideas proyecto* y, posteriormente, en desarrollos concretos. Estas actividades están diseñadas para fomentar la capacidad de los estudiantes de seguir un enfoque estructurado que se base en una metodología clásica de investigación. A través de este proceso, se busca promover el aprendizaje autónomo, continuo y planificado, con un enfoque en las siguientes etapas:

- **Investigación inicial:** búsqueda y análisis de bibliografía actualizada relacionada con el tema.
- **Abstracción del problema:** formulación del desafío como una idea proyecto a resolver.
- **Planificación:** elaboración de una especificación breve del proyecto y definición de un plan de tareas.
- **Diseño:** especificación detallada del uso y las funcionalidades de la aplicación propuesta.

- **Uso crítico de tecnologías:** empleo reflexivo y analítico de herramientas de inteligencia artificial, interpretando sus resultados, identificando limitaciones y sesgos, y evaluando su pertinencia en cada etapa del proyecto.
- **Desarrollo e integración:** implementación del proyecto.
- **Presentación y evaluación:** defensa de la solución en formato oral y escrito.

El proceso es acompañado por los docentes, quienes supervisan, brindan apoyo para resolver dificultades, y revisan el cumplimiento de los lineamientos conceptuales y metodológicos. De este modo, se garantiza que los estudiantes consoliden habilidades clave y logren una experiencia formativa integral.

Se propone un desarrollo grupal integrador que abarca todos los conceptos aprendidos, con un enfoque especial en el proceso de identificación de problemas del mundo real orientado al análisis de datos. Este ejercicio incluye la formulación de dichos problemas como desafíos resolubles desde la perspectiva de la Informática y el diseño de soluciones que puedan ser verificadas y evaluadas en términos de su eficacia y funcionalidad.

Se utilizan herramientas de versionado de código similares a las utilizadas en el ámbito laboral y se promueve el uso de herramientas de software libre y el trabajo colaborativo.

El trabajo integrador incluye el software correspondiente y un informe sobre el mismo. Los trabajos propuestos son aplicaciones interactivas de análisis de datos que presentan una complejidad sencilla y que pueden ser abordados por un estudiante de segundo año.

Para el desarrollo del informe requerido, se realiza un taller en donde se indican las pautas para su correcta redacción. En este taller se trabajan aspectos de redacción, reglas de estilo y formas de incluir citas bibliográficas.

A lo largo de la cursada se introducen aspectos de gamificación, otorgando bonificaciones extras antes diversas actividades **opcionales** propuestas, a las que denominamos Python plus. Estas bonificaciones pueden ser utilizadas luego en algunas de las instancias de evaluación o influir en la nota final de la asignatura.

En las actividades, tanto de teoría como de práctica, se trabaja con los siguientes recursos:
- guías de orientación para los trabajos de producción;
- diapositivas, videos, libros y tutoriales;
- demostraciones de usos de herramientas con ejemplos en vivo;
- el EVEA Moodle como entorno de soporte: https://catedras.linti.unlp.edu.ar/

La interacción con los estudiantes se realiza a través de los mensajes directos o foros provistos por el EVEA y a través de un sistema alternativo tal como Discord.

EVALUACIÓN

A lo largo de la cursada, se proponen diversas actividades que otorgan puntaje, entre las que se incluyen:
1. Desarrollo y entrega de ejercicios en las clases prácticas y teóricas.
2. Exposición y/o explicación de temas.
3. Resolución de desafíos específicos.

Estos **puntos** se distribuyen de acuerdo a los distintos temas abordados en la cursada y tanto la distribución de los mismos como el cronograma tentativo de actividades se publica al comienzo de la misma.

Para la **aprobación** final es necesario:

- **Obtener la cantidad de puntos** establecida en el reglamento de la cursada.
- **Entregar el trabajo integrador** en tiempo y forma, cumpliendo con las especificaciones y etapas de dificultad creciente.
- **Aprobar la defensa individual** del trabajo realizado.
- Realizar una presentación sobre algún contenido de la cursada.

Al final la cursada los estudiantes pueden presentar de forma opciona un informe final donde se evalúa tanto el contenido técnico como la estructura, organización, sintaxis, claridad conceptual y el rigor en la citación de la bibliografía utilizada.

Todas las evaluaciones realizadas son registradas en una planilla detallada, donde se documentan los resultados obtenidos en distintos aspectos: nivel de comprensión, capacidad de aprendizaje, claridad en las presentaciones, organización y expresión en evaluaciones orales, así como la habilidad para resolver desafíos de manera autónoma.

El uso crítico de tecnologías requiere una interpretación consciente de los resultados, reconociendo que las salidas pueden contener limitaciones. Por ello, todo resultado generado por una inteligencia artificial debe ser sometido a validación mediante contraste con fuentes confiables o criterio profesional propio, evitando su adopción acrítica. La interpretación final de las estrategias utilizadas deberán ser fundamentadas por el estudiante, como así también las decisiones de diseño tomadas.

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Fecha	Contenidos/Actividades
1	Semana del 9 de marzo	Presentación de la materia. Conceptos básicos sobre el lenguaje. Tipos de datos básicos. Estructuras de control. Explicación de práctica inicial: explicación sobre el uso de los IDEs propuestos y las pautas para realización de las prácticas. Concepto de software libre.
2	Semana del 16 de marzo	Tipos de datos estructurados: listas, tuplas y diccionarios. Definición de funciones. Taller de git.
3	Semana del 23 de marzo	Funciones con parámetros. **Actividad grupal de práctica.**
4	Semana del 30 de marzo	Expresiones lambda. Introducción al manejo de archivos. **Actividad individual de teoría sobre tipos de datos**.
5	Semana del 6 de abril	Manejo de diferentes tipos de archivos. Conceptos de módulos y paquetes.
6	Semana del 13 de abril	Uso de librería para generación de aplicaciones web. **Actividad individual de práctica.**
7	Semana del 20 de abril	Manejo de excepciones. **Actividad individual de teoría sobre funciones y archivos**.
8	Semana del 27 de abril	Aspectos básicos de programación orientada a objetos en Python. **Actividad individual de teoría sobre manejo de archivos.**

9	Semana del 4 de mayo	Programación orientada a objetos, continuación, Propiedades. **Primera entrega trabajo integrador.**
10	Semana del 11 de mayo	Librerías para el análisis de datos. Presentación del trabajo integrador.
11	Semana del 18 de mayo	Funcionalidades avanzadas de librerías de análisis de datos. Librerías para generación de gráficos. **Actividad individual de teoría sobre excepciones y programación orientada a objetos.**
12	Semana del 25 de mayo	Librería para la generación de aplicaciones web y recursos gráficos interactivos. **Actividad individual de teoría sobre librerías externas y excepciones**
13	Semana de 1 de junio	Aspectos básicos de accesibilidad web y ética informática. Guías para realizar el informe y la presentación final del trabajo integrador.
14	Semana del 8 de junio	Desarrollo del trabajo integrador.
15	Semana del 15 de junio	Desarrollo del trabajo integrador.
16	Semana del 22 de junio	**Entrega segunda parte trabajo integrador.**
17	Semana del 30 de junio	**Actividad integradora individual en la teoría.**
18	Semana del 6 de julio	**Instancias de recuperación.**
19	Semana del 13 de julio	**Instancias de recuperación.**

Contacto de la cátedra (mail, sitio web, plataforma virtual de gestión de cursos):
Mail: python@info.unlp.edu.ar
EVEA: https://catedras.linti.unlp.edu.ar

Firma de la profesora

Prof. Sofía Martin