SEMINARIO DE LENGUAJES (Opción "Python")	**Carrera:** *Licenciatura en Informática* Plan 20*15* *Licenciatura en Sistemas* Plan 20*15* *Analista Programador Universitario* Plan 2015 Analista en TIC Plan 2017
Año 2019	**Año:** 2° **Carácte**r: Electiva
	Duración: Semestral
	Correlatividad: Taller de Programación
	Profesor: Claudia Banchoff, Viviana Harari
	Hs. semanales: 6 hs.

FUNDAMENTACIÓN

Este seminario está orientado a poner en práctica los conceptos vistos en primer año, enfatizando el trabajo sobre la computadora. Los estudiantes reforzarán estos conceptos y aprenderán cómo se los implementa en un lenguaje distinto al utilizado hasta este momento.

OBJETIVOS GENERALES

El objetivo general del Seminario de Lenguaje Python es desarrollar una aplicación concreta, a través de la cual se profundicen los conocimientos obtenidos en los primeros cursos vinculados con algoritmos y programación. Este desarrollo permitirá a los estudiantes llevar a cabo un estudio teórico-práctico del lenguaje de programación Python, poniendo énfasis en el análisis formal de las características del lenguaje y su comparación con los otros lenguajes vistos hasta ese momento.

COMPETENCIAS

- CGS2- Comunicarse con efectividad en forma oral y escrita.
- CGS4- Aprender en forma continua y autónoma, con capacidad de planificar este aprendizaje.
- CGS6- Capacidad para interpretar la evolución de la Informática con una visión de las tendencias tecnológicas futuras.
- CGT1- Identificar, formular y resolver problemas de Informática.
- CGT5- Utilizar de manera efectiva las técnicas y herramientas de aplicación de la Informática.
- LI- CE4– Planificar, dirigir, realizar y/o evaluar proyectos de relevamiento de problemas del mundo real, especificación formal de los mismos, diseño, implementación, prueba, verificación, validación, mantenimiento y control de calidad de sistemas de software/sistemas de información que se ejecuten sobre equipos de procesamiento de datos, con capacidad de incorporación de tecnologías emergentes del cambio tecnológico. Capacidad de análisis, diseño y evaluación de interfases humano computador y computador-computador.
- LS- CE1– Planificar, dirigir, realizar y/o evaluar proyectos de relevamiento de problemas del mundo real. Especificación formal, diseño, implementación, prueba, verificación, validación, mantenimiento y control de calidad de sistemas de software que se ejecuten sobre sistemas de procesamiento de datos, con capacidad de incorporación de tecnologías emergentes del cambio tecnológico. Capacidad de análisis, diseño y evaluación de interfaz humano computador y computador-computador.

CONTENIDOS MÍNIMOS (de acuerdo al Plan de Estudios)

Estudio de un lenguaje de programación en el que se desarrollen aplicaciones concretas. En lo posible la oferta de lenguajes será variable y actualizada con el cambio tecnológico.

PROGRAMA ANALÍTICO

Unidad I. Conceptos de software y recursos libres. El proceso de ejecución de un programa escrito en Python. Características generales.

Unidad II. Sintaxis básica del lenguaje. Tipos predefinidos. Conversiones. Estructuras de control. El formato de un programa de Python. Definición de funciones y módulos. Pasaje de parámetros. Alcances.

Unidad III. Estructuras de datos. Listas, tuplas, conjuntos y diccionarios. Manejo de archivos.

Unidad IV. Abstracción de datos en Python. Conceptos básicos de programación orientada a objetos. Manejo de excepciones.

Unidad V. Resolución de problemas sencillos. Comparación de soluciones a problemas ya vistos destacando las diferencias con los lenguajes de programación conocidos hasta el momento.

Unidad VI. Librerías externas. Generación de gráficos. Procesamiento de formatos típicos como JSON y CVS. Análisis de datos.

Unidad VII. Programación de videojuegos y aplicaciones interactivas. Características generales. Manejo de eventos.

BIBLIOGRAFÏA

- Python Programming: An Introduction to Computer Science. John M. Zelle
- Introduction to Computing and Programming in Python, A Multimedia Approach. Mark Guzdial.
- Python GUI Programming Cookbook. Burkhard A. Meier.
- 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.

METODOLOGÍA DE ENSEÑANZA

La asignatura es de tipo taller, la teoría y práctica se encuentran estrechamente vinculadas. Se trabaja sobre los conceptos aprendidos en las asignaturas de programación que los estudiantes tuvieron hasta esta instancia, comparando y fortaleciendo dichos conceptos pero encarados desde la práctica con el lenguaje Python.

Los teóricos son explicaciones semanales donde se desarrolla, en forma conceptual, cada concepto poniendo énfasis en la capacidad del estudiante para conocer técnicas y herramientas de aplicación en Informática (en lo posible siguiendo las tendencias marcadas por el cambio tecnológico) y en la aplicación efectiva de las mismas. Se acompaña el proceso con materiales para que el estudiante estudie casos y valore la selección y empleo eficiente de herramientas y técnicas determinadas para cada problema. También se plantean actividades en las cuales se analizan tecnologías existentes y "desafía" a los estudiantes a presentar la posible evolución de la solución para ese tipo de problema y en qué podría mejorarse la solución/soluciones actuales.

En los horarios de práctica, se organizan actividades planificadas para los estudiantes, en los que se proponen "desafíos" que deben convertirse en "ideas proyecto" y posteriormente en potenciales desarrollos. Se trata de que el estudiante logre abstraer una serie de pasos que respondan a una metodología clásica de investigación y lo ayuden a aprender en forma continua y autónoma, con capacidad de planificar este aprendizaje:

- Búsqueda de bibliografía actualizada sobre el tema.
- Abstracción del desafío/problema como una "idea proyecto a resolver".
- Expresión sintética de la especificación del proyecto, con recursos humanos requeridos y plan de tareas.
- Implementación y defensa oral/escrita de la solución al desafío.

Se acompaña en todo momento a los estudiantes para que puedan consolidar estas habilidades supervisando, atendiendo las dificultades planteadas y revisando la utilización de los lineamientos conceptuales. Se trabaja con una herramienta de versionado de código que permite introducir buenas prácticas en el trabajo colaborativo

Se plantea un desarrollo final donde se integran todos los conceptos aprendidos poniendo énfasis en el proceso de identificación de problemas del mundo real, especificación de los mismos como problemas resolubles desde la informática y en el desarrollo de soluciones verificables para los mismos. Este trabajo final incluye el software correspondiente y un informe sobre el mismo. Los trabajos propuestos son aplicaciones o juegos educativos para niños y niñas del nivel primario e inicial. Presentan una complejidad sencilla y que pueden ser abordados por una/un estudiante de segundo año.

A lo largo de la cursada se introducen aspectos de gamificación, otorgando puntos extras antes diversas actividades opcionales propuestas, los cuales pueden ser utilizados en algunas de las instancias de evaluación.

Se trabaja con los siguientes recursos:

- Guías de orientación para los trabajos de producción.
- Diapositivas, videos, libros y tutoriales.
- Proyector, computadora, demostraciones de usos de herramientas con ejemplos en vivo.
- EVEA de soporte: *https://catedras.info.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 una canal de Telegram.

EVALUACIÓN

A modo de ejercitación y evaluación se plantean, a lo largo de la cursada, entregas de ejercicios que los estudiantes deben desarrollar y entregar en las prácticas y teorías.

Como evaluación final se plantea un trabajo integrador con entregas obligatorias. Las entregas son acompañadas de coloquios en los cuales los estudiantes deben exponer la tarea realizada y donde el docente puede evaluar no sólo los conocimientos de cada estudiante sino también la claridad de la presentación, su organización y la forma de expresión. La aprobación de las mismas son requisito para la aprobación de la cursada.

Al finalizar la cursada existe una instancia de evaluación final donde los estudiantes deben hacer la entrega de un informe final en comisión y exponer, en forma completa, el trabajo realizado. Para esto se les facilita un lapso de tiempo para que, las comisiones que lo deseen, puedan perfeccionar el trabajo realizado. Posteriormente hay un coloquio grupal donde los docentes interrogan sobre contenidos específicos. Respecto al trabajo escrito se evalúa teniendo en cuenta el contenido técnico, pero también la estructura, organización, sintaxis, claridad conceptual y la bibliografía consultada que debe ser citada rigurosamente.

Toda evaluación realizada a los estudiantes queda plasmada en una planilla muy bien detallada, donde se indican los resultados de las diferentes evaluaciones realizada a los mismos: capacidad del estudiante para desarrollar su aprendizaje, claridad de las presentaciones realizadas, forma de organización y expresión en las diferentes instancias de evaluación oral, formulación de la solución de los diferentes desafíos en forma autónoma, entre otros.

La materia se aprueba con el 75% de los ejercicios prácticos y teóricos aprobados y con el trabajo integrador aprobado.

Se realiza una encuesta sobre los conocimientos iniciales de los estudiantes en la que se releva, además, otra información de interés como ser su situación laboral (evaluación diagnóstica).

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Fecha	Contenidos/Actividades
1	Semana del 12 de marzo	Presentación de la materia. Conceptos básicos. Tipos de datos básicos. Estructuras de datos
2		Explicación de práctica inicial: explicación sobre el uso alguno de los IDEs propuestos y las pautas para realización de las prácticas. Taller de git.
3	Semana del 19 de marzo	Tipos de datos estructurados.
4	Semana del 26 de marzo	Tipos de datos estructurados (continuación).
5	Semana del 9 de abril	Definición de funciones.
6	Semana del 16 de abril	Módulos y paquetes. GUI en Python.
7	Semana del 23 de abril	Manejo de archivos.
8	Semana del 30 de abril	Manejo básico de excepciones.
9	Semana del 7 de mayo	Aspectos básicos de programación orientada a objetos en Python.
10	Semana del 14 de mayo	Especificación del trabajo integrador.
11	Semana del 21 de mayo	Guías para realizar el informe y la presentación final del trabajo integrador.
12	Semana del 28 de mayo	Desarrollo del trabajo integrador.
13	Semana del 3 de junio	Desarrollo del trabajo integrador.
14	Semana del 10 de junio	Desarrollo del trabajo integrador.
15	Semana del 17 de junio	Desarrollo del trabajo integrador.
16	Semana del 24 de junio	Evaluación del trabajo integrador

Evaluaciones previstas	Fecha
Primera evaluación teórico - práctica	Semana del 9 de abril.
Segunda evaluación teórico - práctica	Semana del 7 de mayo
Tercera evaluación teórico - práctica	Semana del 28 de mayo
Cuarta evaluación teórico - práctica	Semana del 11 de junio
Primera evaluación del trabajo integrador	Semana del 18 de junio
Segunda evaluación del trabajo integrador	Semana del 15 de ju|lio
Recuperatorio final	Semana del 6 de agosto

Contacto de la cátedra

Mail: python@info.unlp.edu.ar

EVEAl: catedras.info.unlp.edu.ar

Firma del/los profesor/es