SEMINARIO DE LENGUAJES
(Opción "Python")

Año 2022

Carrera:

Licenciatura en Informática	Plan 2015 / Plan 2021
Licenciatura en Sistemas	Plan 2015 / Plan 2021
Analista Programador Universitario	Plan 2015 / Plan 2021
Analista en TIC	Plan 2017 / Plan 2021

Año: 2°
Carácter: 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 sobre programación 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.

RESULTADOS DE APRENDIZAJE

1.3. Describir los avances informáticos actuales e históricos y demostrar cierta visión sobre tendencias y avances futuros (Básico).
3.1. Definir y diseñar hardware/software informático/de red que cumpla con los requisitos establecidos (Básico).
3.3. Elegir y utilizar modelos de proceso adecuados, entornos de programación y técnicas de gestión de datos con respecto a proyectos que impliquen aplicaciones tradicionales así como aplicaciones emergentes (Básico).
3.4. Describir y explicar el diseño de sistemas e interfaces para interacción persona-ordenador y ordenador-ordenador (Básico).
3.5. Aplicar las correspondientes competencias prácticas y de programación en la creación de programas informáticos y/u otros dispositivos informáticos (Adecuado).
6.1. Organizar su propio trabajo de manera independiente demostrando iniciativa y ejerciendo responsabilidad personal (Básico).
6.3. Planificar su propio proceso de aprendizaje autodidacta y mejorar su rendimiento personal como base de una formación y un desarrollo personal continuos (Básico).

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 interfaces 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. 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. Conceptos básicos de programación orientada a objetos y manejo de excepciones.

Unidad VI. Análisis y uso de librerías externas para la creación de interfaces gráficas de usuario.

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

Unidad VIII. Introducción al análisis y visualización de datos.

BIBLIOGRAFÍA

- 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.
- 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.

METODOLOGÍA DE ENSEÑANZA

Al comienzo de la cursada se realiza una evaluación diagnóstica, a través de una encuesta en línea, que permite conocer tanto los conocimientos iniciales de los estudiantes como información adicional respecto a su situación laboral o aspectos de conectividad.

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.

Las clases teóricas son explicaciones semanales. en donde se desarrolla, en forma conceptual, cada tema 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 evaluar posibles soluciones, presentar la posible evolución de la solución para un mismo problema y realizar comparaciones de eficiencia como así también evidenciar errores que pueden presentarse.

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 y/o problema como una “idea proyecto a resolver”;
- expresión sintética de la especificación del proyecto, y plan de tareas;
- especificación de uso y funcionalidades de la aplicación;
- implementación y defensa, oral y 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 plantea un desarrollo integrador grupal donde se abordan 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. Se utilizan herramientas de versionado de código similares, 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 o juegos educativos 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. Estas bonificaciones pueden ser utilizadas en algunas de las instancias de evaluación.

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 plantean actividades que otorgan puntos. Estas actividades pueden ser ejercicios que los estudiantes deben desarrollar y entregar en las clases prácticas y teóricas, exposición y/o explicación de temas, entre otros. A lo largo de la cursada, los estudiantes deben alcanzar el 70% de

los puntos disponibles, tanto en la teoría como en la práctica. Estos puntos se distribuyen de acuerdo a los distintos temas abordados en la cursada y se publican al comienzo de la cursada.

Otra de las condiciones para la aprobación de la cursada es la entrega en tiempo y en cumplimiento con las especificaciones dadas del trabajo integrador desarrollado, el cual se entrega en distintas etapas, con dificultad creciente. Posteriormente hay un coloquio grupal donde los docentes interrogan sobre contenidos específicos.

Al finalizar la cursada se realiza una instancia de evaluación final donde los estudiantes hacen la entrega de un informe final y exponen, en forma completa, el trabajo realizado. En esta instancia, los grupos de estudiantes que así lo deseen, puedan perfeccionar el trabajo realizado. Respecto al informe escrito se evalúa teniendo en cuenta el contenido técnico, como así 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 detallada, donde se indican los resultados de las diferentes evaluaciones realizadas: el nivel de comprensión del estudiante para desarrollar su aprendizaje, la claridad de las presentaciones realizadas, la forma de organización y expresión en las diferentes instancias de evaluación oral, la formulación de la solución de los diferentes desafíos en forma autónoma, entre otros.

La materia se aprueba obteniendo al menos el 70% de los puntos (en la teoría y en la práctica) y el trabajo integrador aprobado.

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Fecha	Contenidos/Actividades
1	Semana del 7 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 14 de marzo	Tipos de datos básicos. Estructuras de control (cont.). Taller de git.
3	Semana del 21 de marzo	**Actividad grupal en las prácticas.**
4	Semana del 28 de marzo	Tipos de datos estructurados: listas, tuplas y diccionarios. Definición de funciones.
5	Semana del 4 de abril	**Actividad individual en la teoría sobre tipos estructurados**. Funciones con parámetros. Expresiones lambda. Introducción a una librería gráfica.
6	Semana del 11 de abril	Conceptos de módulos y paquetes. Uso de la librería gráfica. **Actividad individual en las prácticas.**
7	Semana del 18 de abril	Manejo de archivos.
8	Semana del 25 de abril	Manejo de excepciones.
9	Semana del 2 de mayo	**Actividad individual en la teoría sobre funciones y archivos**. Aspectos básicos de programación orientada a objetos en Python.
10	Semana del 9 de mayo	Desarrollo del trabajo integrador.
11	Semana del 16 de mayo	Iteradores y generadores. Presentación del trabajo integrador.

12	Semana del 23 de mayo	Desarrollo del trabajo integrador.
13	Semana de 30 de mayo	**Actividad individual en la teoría sobre excepciones y POO.** Desarrollo del trabajo integrador.
14	Semana del 6 de junio	Guías para realizar el informe y la presentación final del trabajo integrador. Taller en clase. **Primera entrega trabajo integrador.**
15	Semana del 13 de junio	Taller sobre ética en Informática. **Actividad integradora individual en la teoría.**
16	Semana del 20 de junio	Desarrollo del trabajo integrador.
17	Semana del 27 de junio	Desarrollo del trabajo integrador.
18	Semana del 4 de julio	**Segunda entrega trabajo integrador.**

Contacto de la cátedra

Mail: python@info.unlp.edu.ar

EVEA: https://catedras.linti.unlp.edu.ar

Firma de las profesoras

Prof. Claudia Banchoff

Prof. Harari Viviana