Ministerio de Cultura y Educación **(Programa del año 2021)**
Universidad Nacional de San Luis
Facultad de Ciencias Físico Matemáticas y Naturales
Departamento: Informatica
Area: Area V: Automatas y Lenguajes

I - Oferta Académica

Materia	Carrera	Plan	Año	Período
DISEÑO Y PARADIGMAS DE LENGUAJES	ING. INFORM.	026/12-08/15	2021	2° cuatrimestre
DISEÑO Y PARADIGMAS DE LENGUAJES	ING. EN COMPUT.	28/12	2021	2° cuatrimestre

II - Equipo Docente

Docente	Función	Cargo	Dedicación
CAGNINA, LETICIA CECILIA	Prof. Responsable	P.Adj Exc	40 Hs
ROGGERO, PATRICIA BEATRIZ	Prof. Co-Responsable	P.Asoc Exc	40 Hs
FUNEZ, DARIO GUSTAVO	Responsable de Práctico	JTP Exc	40 Hs
LOOR, FABRICIO	Auxiliar de Práctico	A.1ra Simp	10 Hs

III - Características del Curso

Credito Horario Semanal				
Teórico/Práctico	**Teóricas**	**Prácticas de Aula**	**Práct. de lab/ camp/ Resid/ PIP, etc.**	**Total**
Hs	2 Hs	2 Hs	2 Hs	6 Hs

Tipificación	Periodo
B - Teoria con prácticas de aula y laboratorio	2º Cuatrimestre

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
23/08/2021	26/11/2021	14	75

IV - Fundamentación

Las prácticas de programación desarrolladas en cursos previos como Programación I y Programación II, brindan al futuro ingeniero una experiencia concreta en la resolución de problemas utilizando ciertos lenguajes de programación. Esta experiencia brinda también al alumno una primera aproximación a distintos aspectos y construcciones de los lenguajes de programación utilizados, como así también a los paradigmas a los cuales estos pertenecen.
Acotar el conocimiento de un ingeniero a un paradigma o lenguaje particular, acota su visión de las herramientas existentes a las particularidades del lenguaje y paradigma utilizados. Esto no responde en general a las necesidades actuales, con un mercado laboral que plantea problemas de diversa índole, cuya resolución efectiva puede requerir de construcciones y paradigmas de lenguajes diversos. Todo esto además, en un contexto donde permanentemente se le ofrecen al profesional informático nuevos lenguajes (o adaptaciones de los ya existentes) y paradigmas que intentan dar respuesta a las necesidades que los nuevos tipos de aplicaciones demandan.

Es así que surge la necesidad de brindar al futuro ingeniero una visión más global de los lenguajes, que profundice más allá de la consideración superficial de sus “características” y se exploren los principales conceptos de diseño subyacentes y su efecto sobre la implementación de los lenguajes. Esta visión, de acuerdo a la bibliografía especializada en el tema permite, entre otras cosas, mejorar la habilidad para desarrollar algoritmos eficaces, mejorar el uso del lenguaje de programación

disponible, acrecentar el propio vocabulario con construcciones útiles de programación, hacer posible una mejor elección del lenguaje de programación y facilitar el aprendizaje de un nuevo lenguaje. Si a todo esto se le suma la identificación de los principios subyacentes de los principales paradigmas de lenguajes de programación, y una comparación crítica entre los mismos, se puede decir que se le brindan al estudiante las herramientas necesarias para enfrentar sus necesidades presentes y futuras a la hora de elegir y usar de manera adecuada un lenguaje de programación.

V - Objetivos / Resultados de Aprendizaje

Al finalizar el curso se espera que el alumno sea capaz de:

• Tener una perspectiva general de los paradigmas claves que se usan en el desarrollo de lenguajes de programación modernos, sus bases teóricas, aplicativas y de implementación: lenguajes Imperativos, Funcionales, Lógicos y Orientados a Objetos.

• Desarrollar una visión clara del tipo de situaciones en que los distintos paradigmas son adecuados y hacer uso de lenguajes multi-paradigma que permitan una fácil integración de los mismos y su interacción con lenguajes de programación existentes.

• Evaluar en forma crítica distintos lenguajes de programación existentes y futuros.

• Entender la implementación de distintos lenguajes con suficiente detalle como para reconocer la relación entre un programa fuente y su comportamiento en ejecución.

• Extender sus conocimientos sobre los temas anteriores con bibliografía adecuada y mínima supervisión.

VI - Contenidos

UNIDAD I.
Razones del estudio de lenguajes de programación. Historia de los lenguajes de programación. Características de un buen lenguaje. La estructura y operación de una computadora. Computadora de hardware y firmware. Traductores e Intérpretes. Computadoras virtuales y ligaduras. Tiempos de ligadura.

UNIDAD II.
Sintaxis y semántica de los lenguajes. Concepto de Objeto de Datos. Declaraciones. Utilidad y propósitos de las declaraciones. Operaciones. Chequeo de tipos: estático y dinámico. Conversión de tipos. Especificación e implementación de tipos de datos elementales y estructurados.

UNIDAD III.
Definición y activación de subprogramas. Administración de la memoria. Fases de la administración de la memoria. Administración de memoria estática. Administración de memoria basada en pila. Administración de memoria basada en heap con elementos de tamaño fijo y de tamaño variable. Mecanismos de recuperación. Ejemplos.

UNIDAD IV.
Control de subprogramas. Llamada-retorno simple. Subprogramas recursivos. Control de datos. Ambientes de referenciación. Alcance estático y dinámico. Datos compartidos en subprogramas. Parámetros. Pasaje de parámetros. Ambientes comunes explícitos. Alcance dinámico, reglas e implementaciones. Alcance estático, reglas e implementaciones.

UNIDAD V.
Variantes en el control de subprogramas. Excepciones y manejadores de excepciones. Eventos y manejadores de eventos. Corutinas. Subprogramas planificados. Programación paralela: Comandos en guardia. Tareas.

UNIDAD VI.
Accediendo al hardware en distintos lenguajes de programación. Introducción a la conexión de dispositivos de E/S. Dispositivos de E/S. Placas Arduino o similares. Características de Hardware y Software de las placas. Conexión placa-pc. Distintas componentes de entrada/salida para incorporar a las placas. Ejemplos de aplicaciones utilizando placas y componentes.

UNIDAD VII.
Paradigmas de lenguajes: imperativo, funcional, orientado a objetos y lógico. Lenguajes de programación multiparadigma. Definición. Utilidad. Paradigma de Programación Orientada a Objetos: tipos de datos abstractos, herencia, polimorfismo. Diferentes aspectos del diseño de los lenguajes orientados a objetos: Smalltalk, C++ y Java.

VII - Plan de Trabajos Prácticos

Los prácticos de la asignatura son del tipo:

a) Prácticos de aula:

Práctico 1: Tipos de Datos Elementales y Estructurados.
Objetivos: ejercitar sobre operaciones, signatura, declaraciones, tipos de datos elementales y chequeo de tipos. Analizar y ejercitar sobre la especificación e implementación de tipos de datos estructurados.
Metodología: resolución de ejercicios en lápiz y papel. Implementar algunos de los ejercicios en máquina.

Práctico 2: Administración de Memoria.
Objetivos: ejercitar definición y activación de subprogramas, mecanismos de administración de memoria, problemas asociados al uso de punteros y métodos de recuperación.
Metodología: resolución de ejercicios en lápiz y papel. Implementar algunos de los ejercicios en máquina.

Práctico 3: Control de Secuencia y Datos en Subprogramas.
Objetivos: realizar ejercicios referidos a ambientes de referenciación, implementación de reglas de alcance estático y dinámico.
Metodología: resolución de ejercicios en lápiz y papel.

Práctico 4: Variantes en el control de subprogramas.
Objetivos: desarrollo de ejercicios con variantes en control de subprogramas: excepciones, eventos y concurrencia en Java.
Metodología: ejercicios desarrollados en máquina.

Práctico 5: Dispositivos de Entrada/Salida.
Objetivos: programar ejercicios para visualizar cómo se realiza la entrada/salida de diferentes dispositivos usando una placa tipo Arduino.
Metodología: resolución de ejercicios e implementación en máquina utilizando placas tipo Arduino o algún simulador de la mismas.

b) Prácticos de formación experimental:

Laboratorio 1: Implementación de alguno de los métodos de recuperación de la memoria.
Objetivos: desarrollo y prueba de algún algoritmo donde se utilicen e implementen facilidades de administración de memoria utilizando el lenguaje Java.
Metodología: resolución en computadora que permita visualizar los conceptos mencionados.

VIII - Regimen de Aprobación

Los contenidos de la materia impartidos durante el periodo no presencial (clases virtuales), así como las evaluaciones que se realicen, tendrán validez académica según estipula el Art. 7 de la Resol. CD 018-20. Las clases virtuales se impartirán mediante el uso de tecnologías como Google Meet, Zoom, videos preparados por la cátedra y material escrito que se proveerá a través de Google Classroom, página web de la materia y/o aulas virtuales (como Moodle). Por tal motivo, no se considera la asistencia en el régimen de aprobación de la materia.

No se admite rendir la materia en condición de libre. El alumno puede optar por alguna de las dos alternativas del régimen de

aprobación: A. Regularizar la materia y luego rendir un examen final; B. Promocionar la materia, cumpliendo las condiciones que se detallan a continuación.

A. Para regularizar la materia, el alumno deberá:

A1. Entregar el 100% de las evaluaciones periódicas solicitadas y tener aprobadas al menos el 80% de ellas. Estas evaluaciones consistirán de la entrega de ejercicios de prácticos de aula resueltos, y/o entrega de ejercicios extras propuestos por los docentes, y/o entrega de trabajos de desarrollo tipo informes. Los enunciados de las evaluaciones serán provistos en la plataforma Classroom y/o aulas virtuales de la materia. Al menos habrá una entrega por cada Unidad desarrollada.

A2. Aprobar los prácticos de laboratorio propuestos.

B. Para promocionar la materia, el alumno deberá:

B1. Cumplir con los requerimientos A1 y A2 enunciados anteriormente.

B2. Aprobar con un mínimo de 7 (siete) un examen integrador oral y/o escrito al final del cuatrimestre bajo la modalidad que se comunique oportunamente (Arts. 10 y 11 de la Resol. CD 018-20).

B3. La nota final de la materia se computará promediando las notas obtenidas en cada uno de los puntos mencionados previamente.

IX - Bibliografía Básica

[1] "Programming Languages - Design and Implementation". Pratt, Terrence y Zelkowitz, Marvin. Cuarta edición. Prentice Hall, 2001.
[2] "Lenguajes de Programación - Diseño e Implementación". Pratt, Terrence y Zelkowitz, Marvin. Tercera edición. Prentice Hall, 1999.
[3] "Concepts of Programming Languages". Sebesta, Robert. Addison-Wesley. Ediciones 2004, 2016 y 2019.
[4] "El lenguaje de Programación C". Kernighan, Brian y Ritchie, Dennies. Prentice Hall, 1991.
[5] Apunte de la cátedra: "Sintaxis, Traducción, Fases de la Compilación".
[6] Apunte de la cátedra: "Evolución del Concepto de Tipo de Datos-Tipo de Datos Abstractos-Programación Orientada a Objetos".

X - Bibliografia Complementaria

[1] "Smalltalk-80. The Language and its implementation". Goldberg, Adele y Robson, David. Addison-Wesley, 1985.
[2] The Java class libraries. Chan, Patrick - Lee, Rosanna y Kramer, Douglas. Addison Wesley. Segunda Edición, 1998.
[3] "Piensa en Java". Eckel Bruce. Pearson Alhambra. Cuarta Edición, 2007.

XI - Resumen de Objetivos

El curso tiene como objetivo introducir al alumno a la problemática del diseño e implementación de lenguajes de programación, incluyendo fundamentos teóricos y prácticos. El estudio se realiza teniendo en cuenta todos los paradigmas actuales de programación, realizando un estudio comparativo de las técnicas de implementación de cada uno de ellos.

XII - Resumen del Programa

Historia de los lenguajes de programación. Evolución de los paradigmas de programación. Computadoras virtuales. Características esenciales de los lenguajes de programación y su implementación: tipos de datos y su representación, control de secuencia y datos en subprogramas. Administración de Memoria. Abstracción de Datos. Variantes en el control de subprogramas. Programación Multi-paradigma.

XIII - Imprevistos

A los efectos de que se impartan todos los contenidos mínimos de la materia y se respete el crédito horario establecido en el

plan de estudios de la carrera para esta asignatura, se establece que se dicten cómo máximo 6hs por semana distribuidas en teorías, prácticos de aula, laboratorios y consultas, hasta completar las 75hs del crédito total.
El dictado de las clases teóricas será mediante videoconferencias en plataformas tipo Google Meet y apoyadas con TICs.
Los laboratorios se realizarán en computadoras individuales mediante el uso de lenguajes de programación específicos y el uso de simuladores.
El presente programa puede presentar ajustes dada la situación epidemiológica por la pandemia COVID-19. Toda modificación será acordada y comunicada con el estudiantado e informada a Secretaría Académica.
Información de contacto: Dra. Leticia Cagnina lcagnina@email.unsl.edu.ar.

XIV - Otros