Ministerio de Cultura y Educación **(Programa del año 2017)**
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	2017	2° cuatrimestre
DISEÑO Y PARADIGMAS DE LENGUAJES	ING. EN COMPUT.	28/12	2017	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.Adj 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	1 Hs	2 Hs	5 Hs

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

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
07/08/2017	18/11/2017	15	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 a los principales paradigmas en lenguajes de programación, y una comparación crítica entre los mismos, se puede decir que se le brindan al Ingeniero, 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, de firmware. Traductores e Intérpretes. Computadoras virtuales y ligaduras. Tiempos de ligadura.

UNIDAD II.
Sintaxis de los lenguajes de programación. Criterios sintácticos generales. Métodos para la descripción de la sintaxis y semántica de los lenguajes. Chequeo de tipos: estático y dinámico. Declaraciones. Operaciones. Conversión de tipos. Equivalencia 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 memoria. Administración de memoria estática. Administración de memoria basada en pila. Heap con elementos de tamaño fijo y de tamaño variable. Mecanismos de recuperación. Ejemplos.

UNIDAD IV.
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. Distintos componentes de entrada/salida para incorporar a las placas. Ejemplos de aplicaciones utilizando placas y componentes (en Laboratorio).

UNIDAD V.
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 VI.
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 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: Aspectos del Diseño de Lenguajes.
Objetivos: visualizar las características de un buen lenguaje, ejercitar sobre los distintos tipos de ligaduras.
Metodología: resolución de ejercicios en lápiz y papel.

Práctico 2: 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 los tipos de datos estructurados.
Metodología: resolución de ejercicios en lápiz y papel. Implementar algunos de los ejercicios en máquina.

Práctico 3: 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 4: 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.

Práctico 5: 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. Implementar algunos de los ejercicios en máquina.

Práctico 6: 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.

b) Prácticos de formación experimental:

Laboratorio 1: Aplicaciones de Entrada/Salida.
Objetivos: desarrollo de ejercicios donde se utilicen e implementen facilidades de E/S en placas tipo Arduino.
Metodología: resolución de ejercicios en computadora que permitan analizar y comparar los conceptos mencionados. Elaboración y entrega de informe.

VIII - Regimen de Aprobación

El alumno puede regularizar y luego rendir un examen final, o promocionar la materia, cumpliendo las siguientes condiciones:

A. Régimen para alumnos Regulares:

1) Tener un mínimo de 70% de asistencia a las clases prácticas y teóricas.

2) Aprobar los prácticos de laboratorio.

3) Tener al menos el 80% de los ejercicios de prácticos de aula aprobados que sean solicitados por la cátedra.

4) Aprobar dos exámenes parciales que incluirán temas de todos los prácticos, o alguna de sus respectivas recuperaciones con al menos el 70% correcto del total.
Tal como lo establece la reglamentación vigente, Ord. 32/14 CS, cada examen parcial tendrá dos (2) recuperaciones.

B. Régimen para alumnos Promocionales:

1) Ídem a lo requerido para alumnos Regulares, salvo que el alumno deberá asistir al 80% de las clases tanto teóricas como prácticas.

2) Aprobar con un mínimo de 7 (siete) un examen integrador oral y/o escrito al final del cuatrimestre.

3) La nota final se computará promediando las notas obtenidas en cada uno de los puntos mencionados previamente.

C. Para aprobar la materia, los alumnos regulares deberán rendir un examen final, el cual podrá ser oral y/o escrito.

D. No se admite rendir la materia en condición de libre.

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. Edición, 2004 y Edición, 2016.
[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

XIV - Otros