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
ANALISIS COMPARATIVO DE LENGUAJES	LIC.CS.COMP.	18/11	2021	2° cuatrimestre
ANALISIS COMPARATIVO DE LENGUAJES	PROF.CS.COMPUT.	02/16	2021	2° cuatrimestre
ANALISIS COMPARATIVO DE LENGUAJES	LIC.CS.COMP.	006/05	2021	2° cuatrimestre
ANALISIS COMPARATIVO DE LENGUAJES	LIC.CS.COMP.	32/12	2021	2° cuatrimestre
ANALISIS COMPARATIVO DE LENGUAJES	PROF.CS.COMPUT.	06/09	2021	2° cuatrimestre

II - Equipo Docente

Docente	Función	Cargo	Dedicación
ROGGERO, PATRICIA BEATRIZ	Prof. Responsable	P.Asoc Exc	40 Hs
ERRECALDE, MARCELO LUIS	Prof. Colaborador	P.Asoc Exc	40 Hs
FUNEZ, DARIO GUSTAVO	Responsable de Práctico	JTP Exc	40 Hs
GATICA, CLAUDIA RUTH	Auxiliar de Práctico	A.1ra Semi	20 Hs

III - Características del Curso

Credito Horario Semanal				
Teórico/Práctico	**Teóricas**	**Prácticas de Aula**	**Práct. de lab/ camp/ Resid/ PIP, etc.**	**Total**
Hs	3 Hs	4 Hs	2 Hs	9 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	120

IV - Fundamentación

Debido al contexto epidemiológico actual, en concordancia a las medidas adoptadas a nivel nacional, provincial y por la UNSL, debido a la pandemia provocada por el COVID-19, este programa se desarrollará en MODALIDAD MIXTA (PRESENCIAL Y NO PRESENCIAL), utilizando para tal fin diferentes herramientas virtuales, para el dictado de clases teóricas, clases de práctica, consultas y evaluaciones.

Los conceptos involucrados en el estudio del diseño y paradigmas de lenguajes de programación, forman parte del conocimiento general de un Licenciado en Ciencias de la Computación, con el perfil requerido por los correspondientes planes de estudios.

Esta asignatura complementa los contenidos introducidos en las materias Programación I y Programación II, donde el énfasis está puesto en la resolución de problemas utilizando diferentes lenguajes de programación, esto brinda al/la estudiante 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.

Circunscribir el conocimiento de un Licenciado en Ciencias de la Computación 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 cambiante, que plantea de forma continua problemas de diversa naturaleza, cuya resolución efectiva puede requerir de construcciones y lenguajes diversos.

Es por esto que resulta necesario brindar al futuro/a Licenciado/a una visión global de los lenguajes, donde se exploren los principales conceptos de diseño subyacentes y su efecto sobre la implementación de los lenguajes. Esta visión 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 brinda a los/las estudiantes las herramientas necesarias para enfrentar sus necesidades presentes y futuras a la hora de elegir y usar de manera adecuada un lenguaje de programación.

Además, en el convencimiento de que los aspectos básicos del diseño de compiladores deberían formar parte del conocimiento general de cualquier buen programador/a, este curso introduce la estructura básica de un compilador. Además, se abordan aspectos relacionados a la teoría formal de lenguajes que son necesarios para entender el funcionamiento de un compilador, pero servirán además como introducción a tópicos más avanzados a desarrollarse en otros cursos de la carrera como Autómatas y Lenguajes, y Diseño y Construcción de Compiladores.

Los contenidos del curso se corresponden con las unidades de conocimiento recomendadas por la ACM/IEEE Computer Society Joint Curriculum Task Force, para el área de lenguajes de programación.

V - Objetivos / Resultados de Aprendizaje

Se presentan a continuación una serie de objetivos tanto de carácter general o transversal como de carácter específico, que se pretende que alcancen los/las estudiantes a lo largo del dictado del curso.

Objetivos generales
- Promover el análisis crítico de los temas abordados.
- Promover el autoconocimiento y la autonomía.
- Promover el trabajo grupal y colaborativo.
- Promover la expresión oral y escrita.

Objetivos específicos:
- Evaluar en forma crítica distintos lenguajes de programación existentes y futuros.
- Responder a cuestiones tales como las motivaciones de la existencia de tantos lenguajes de programación, cómo y porqué fueron desarrollados, en qué se asemejan y difieren.
- Reconocer los paradigmas claves usados en el desarrollo de lenguajes de programación modernos, sus bases teóricas, aplicativas y de implementación.
- Entender la implementación de distintos lenguajes, como para reconocer la relación entre un programa fuente y su comportamiento en ejecución.
- Extender sus conocimientos sobre los tópicos anteriores con bibliografía adecuada y mínima supervisión.
- Realizar un curso sobre el estudio del diseño y construcción de compiladores.

VI - Contenidos

UNIDAD 1. Características del Diseño de Lenguajes de Programación
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. Evolución de Paradigmas de Programación.

UNIDAD 2. Programación Lógica
Programación Lógica. El lenguaje Prolog. Elementos básicos: hechos, predicados y consultas. Átomos y variables. Variables anónimas. Matching. Reglas recursivas. Proceso de backtracking. Functores. Listas.

UNIDAD 3. Aspectos Formales de los Lenguajes de Programación
Sintaxis de los lenguajes de programación. Criterios sintácticos generales. Nociones básicas de semántica formal. Elementos sintácticos de un lenguaje. Estructura programa-subprograma completo. Etapas en la traducción. Modelos de traducción formales. Lenguajes Regulares y Lenguajes Libres del Contexto. Autómatas Finito. Expresiones Regulares. Gramáticas Libres del Contexto. BNF. BNFE. Autómatas finitos. Expresiones regulares.

UNIDAD 4. Tipos de Datos
Propiedades de tipos y objetos. Tipos de datos. Especificación e implementación de tipos de datos elementales. Operaciones. Declaraciones. Chequeo de tipos. Tipos de datos numéricos. Enumeraciones. Booleanos. Caracteres. Tipos de datos estructurados. Especificación e implementación. Vectores y arreglos. Registros. Cadenas de caracteres. Punteros. Conjuntos. Control de secuencia explícito e implícito. Secuenciamiento en expresiones aritméticas. Representación de árbol.

UNIDAD 5. Administración de Memoria
Definición y activación de subprogramas. Administración de 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.

UNIDAD 6. Control de Datos en Subprogramas
Control de subprogramas. Llamada-retorno simple. La regla de la copia. Subprogramas recursivos. Control de datos. Ambientes de referenciación. Alcance estático y dinámico. Estructura de bloques. Datos compartidos en subprogramas. Parámetros. Pasaje de parámetros. Ambientes comunes explícitos. Alcance dinámico, reglas e implementaciones. Alcance estático, reglas e implementaciones.

UNIDAD 7. Variantes del Control de Subprogramas
Variantes en control de subprogramas. Excepciones y manejadores de excepciones. Corutinas. Subprogramas planificados. Programación paralela: comandos en guardia, tareas.

UNIDAD 8. Programación Orientada a Objetos
Tipos de datos abstractos. Evolución del concepto de tipo de datos. Ocultamiento de la información. Encapsulamiento mediante subprogramas. Programación orientada a objetos: herencia, polimorfismo y ligadura dinámica. Subclases y subtipos. Herencia simple y múltiple. Aspectos de diseño en Smalltalk, C++ y Java.

VII - Plan de Trabajos Prácticos

Práctico 1: Prolog (aula y formación experimental).
Objetivos: comprender las características de un lenguaje lógico, aprender los aspectos claves del lenguaje Prolog, reglas de matching, recursión y estructuras.
Metodología: resolución de ejercicios en lápiz y papel, luego ejecutar dichos ejercicios en máquina.

Práctico 2: Aspectos del diseño de lenguajes de programación.
Objetivos: visualizar las características de un buen lenguaje, computadora virtual, ejercitar sobre los distintos tipos de ligaduras.
Metodología: análisis y resolución de ejercicios.

Práctico 3: Descripción Formal de Lenguajes.
Objetivos: ejercitar expresiones regulares, autómatas finitos, gramáticas libres del contexto, BNF y BNFE.
Metodología: resolución de ejercicios en lápiz y papel.

Práctico 4: Tipos de Datos Elementales y Estructurados.
Objetivos: ejercitar sobre operaciones, signatura, declaraciones, tipos de datos elementales y estructurados. Chequeo de tipos.
Metodología: resolución de ejercicios en lápiz y papel.

Práctico 5: 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.

Práctico 6: Control de Datos en Subprogramas.
Objetivo: realizar ejercicios referidos a ambientes de referenciación, implementación de reglas de alcance estático y dinámico.
Metodología: resolución de ejercicios en lápiz y papel.

Práctico 7: Variantes en control de subprogramas y POO.
Objetivo: desarrollo de ejercicios con variantes en control de subprogramas: excepciones, corutinas, subprogramas planificados y POO.
Metodología: resolución de ejercicios en lápiz y papel.

Prácticos de Formación Experimental:
Laboratorio 1: Lenguaje Prolog. Realización de un proyecto utilizando el lenguaje.
Objetivos: evaluar consultas, aplicación de las reglas de matching, programación del proyecto.
Metodología: resolución en computadora de escritorio.

Laboratorio 2: Uso de Expresiones Regulares
Objetivos: utilización de una herramienta que realiza búsquedas de patrones empleando expresiones regulares, sobre un conjunto de documentos de textos suministrado por la cátedra.
Metodología: resolución en computadora de escritorio.

VIII - Regimen de Aprobación

La propuesta de trabajo, en lo que respecta al régimen de aprobación, está basada en la evaluación continua o formativa, con el objetivo de relacionar la información sobre la evolución del proceso de aprendizaje de los/las estudiantes con las características de la acción didáctica, a medida que se desarrollan y progresan las actividades de enseñanza y aprendizaje.

En este sentido, el/la estudiante puede regularizar (para luego rendir el examen final) o promocionar. Las condiciones en función a la modalidad de dictado mixta (presencial y no presencial) y a la metodología propuesta en la que se desarrollará el curso, son las siguientes:

A. Régimen para Estudiantes Regulares
Entregar, en tiempo y forma, y aprobar el 100% de las evaluaciones periódicas propuestas por la cátedra, las cuales consistirán en ejercicios de los prácticos de aula, y/o ejercicios prácticos particulares propuestos por los docentes, y/o prácticos de laboratorio, y/o trabajos de investigación, que pueden incluir exposición, de temas o problemas particulares sobre los contenidos de la materia.
Con la finalidad de realizar un trabajo colaborativo de comprensión, se efectuarán devoluciones de las instancias evaluativas como parte del proceso de aprendizaje. Además, todas las evaluaciones pueden tener una instancia de defensa oral posterior a la entrega donde el/la estudiante exponga, explique y responda dudas o inquietudes de sus compañeros y/o docentes acerca del trabajo realizado en sus entregas. También se podrán incluir actividades de autoevaluación.
Los/las estudiantes deben contar además con un porcentaje mínimo igual o superior al 60% de participación activa en todas las actividades sincrónicas y asíncronas propuestas por los docentes durante el cursado de la materia.
Aclaración::
1. Las evaluaciones se aprueban con un porcentaje mínimo de setenta por ciento (70%).
2. Si cualquier punto no fuera cumplimentado implica que el/la estudiante pase a condición de libre.

B. Régimen para Estudiantes Promocionales
1. Ídem a lo requerido para estudiantes regulares excepto que se debe alcanzar un porcentaje de asistencia igual o superior al

80%.
2. Además el estudiante tendrá que desarrollar y aprobar un coloquio integrador oral o escrito que incluya el análisis y discusión de conceptos teóricos dados en el curso. Dicho coloquio podrá ser de forma virtual o presencial (en el caso que estén dadas las condiciones para ello).
La nota final de promoción se computará promediando, las notas obtenidas en las evaluaciones requeridas para la regularidad y la nota obtenida en el ítem 2 del punto B. En todas las evaluaciones, la nota obtenida por el alumno debe ser igual a 7 o superior, según lo establece la normativa vigente.

C. El curso no admite rendir el examen final en condición de Libre.

D. El examen final puede ser oral y/o escrito.

IX - Bibliografía Básica

[1] "Programming Languages - Design and Implementation". Pratt, Terrence y Zelkowitz, Marvin. Cuarta edición. Prentice Hall, 2001.
[2] "Concepts of Programming Languages". Sebesta, Robert. Addison-Wesley. Edición 2019 y Edición 2016.
[3] "Prolog, programming for artificial intelligence". Bratko, Ivan. Addison-Wesley. Tercera Edición, 2001.
[4] "El lenguaje de Programación C". Kernighan, Brian y Ritchie, Dennies. Prentice Hall, 1991.
[5] Apunte de la Cátedra Aspectos formales de la traducción de lenguajes
[6] Apunte de la Cátedra de Evolución del Concepto de Tipos de Datos - Tipo de Dato Abstracto - Programación Orientada a Objetos.
[7] Apunte de la Cátedra del Lenguaje Prolog.

X - Bibliografia Complementaria

[1] "Lenguajes de Programación - Diseño e Implementación". Pratt, Terrence y Zelkowitz, Marvin. Tercera edición. Prentice Hall, 1999.
[2] "Smalltalk-80. The Language and its implementation". Goldberg, Adele y Robson, David. Addison-Wesley, 1985.
[3] The Java class libraries. Chan, Patrick - Lee, Rosanna y Kramer, Douglas. Addsion Wesley. Segunda Edición, 1998.
[4] "Piensa en Java". Eckel Bruce. Pearson Alhambra. Cuarta Edición, 2007.
[5] "Introduction to Java". Carlos Kavka. ICTP, 2004.
[6] "Introduction to Java". Hume, J.N. Patterson y Stephenson, Christine. Canada Holt Software Associates. Primera Edición, 2000.

XI - Resumen de Objetivos

El curso tiene como objetivo introducir al alumno a la problemática del diseño e implementación de lenguajes, incluyendo fundamentos teóricos y modelos formales. 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. Sistemas de traducción. Sintaxis y semántica. Gramáticas, expresiones regulares, autómatas. Características esenciales de los lenguajes de programación y su implementación: tipos de datos y su representación, control de secuencia y datos. Administración de Memoria. Abstracción de Datos. Programación lógica, lenguaje Prolog. Control de datos a nivel de subprogramas. Variantes en el control de subprogramas.

XIII - Imprevistos

El presente programa puede presentar ajustes y/o cambios provocados por la situación epidemiológica por COVID19. Toda modificación será acordada y comunicada con el estudiantado e informada a Secretaría Académica.
Aclaración: de acuerdo a la normativa vigente respecto al Calendario Académico de la Universidad Nacional de San Luis, el segundo cuatrimestre tendrá una duración de 14 semanas. A los efectos de que se impartan todos los contenidos y se respete el crédito horario establecido en el Plan de Estudios de la carrera para esta asignatura, se establece que se de cómo máximo 9

horas por semana distribuidas en teorías, prácticos de aula y laboratorio, consultas, hasta completar las 120 horas correspondientes al crédito horario total de la asignatura.

XIV - Otros