Cátedra de la Carrera Licenciatura en Sistemas de Información de la Facultad de Ciencia y Tecnología de la UADER

CATEDRA: Paradigmas y Lenguajes

EQUIPO DOCENTE: Esp. Ing. Claudia Alvarez

AÑO ACADEMICO: 2019

PLAN DE ESTUDIO: 2010

AÑO DE LA CARRERA A LA QUE PERTENECE LA CATEDRA: Tercero

RÉGIMEN DE LA MATERIA: cuatrimestral

CARGA HORARIA SEMANAL: 4

1- FUNDAMENTACION

Dentro de las actividades del Licenciado en Sistemas de Información, el dominio de lenguajes de programación es de suma importancia como conocimiento base para el análisis y diseño de sistemas y como fundamento para el desarrollo de software. En ambos casos es necesario comprender los modelos formales en los cuales se basa la definición léxica, sintáctica y semántica de los lenguajes de programación y los estilos de programación.
La definición de lenguajes permite formalizar instrumentos para el análisis y también posibilita la creación de dialectos aplicados a cálculo de sueldos, robótica, simulación, procesamiento del lenguaje natural, anti virus, etc.
También se hace una introducción a los paradigmas de programación lo cual contribuye a la formación en las Ciencias de la Computación

2- OBJETIVOS

Generales: Que el alumno comprenda los conceptos relativos a los modelos formales en los cuales se basa la definición léxica, sintáctica y semántica de los lenguajes de programación y los estilos de programación

Objetivos Específicos:

Unidad 1: Comprender los conceptos básicos relativos a Traductores, Compiladores, Intérpretes.

Unidad 2: Comprender los conceptos relativos a los lenguajes formales.

Unidad 3: Comprender los conceptos sobre Autómatas finitos.

Unidad 4: Comprender Expresiones Regulares. Introducción a las Gramáticas y Máquinas de Turing

Unidad 5: Introducción a los paradigmas de programación.

3- PROGRAMA DE CONTENIDOS

a) Contenidos mínimos propuestos en el plan de estudios

Semántica y sintaxis de lenguajes. Lenguajes formales y autómatas. Minimización de Autómatas. Expresiones Regulares. Máquinas de Turing. Jerarquía de Chomsky. Gramáticas e Isomorfismos. Lenguajes de Programación: Entidades y ligaduras. Sistemas de Tipos. Niveles de polimorfismo. Encapsulamiento y Abstracción. Intérpretes y Compiladores. Criterios de Diseño y de Implementación de Lenguajes de Programación. Nociones básicas de semántica formal. Paradigmas de lenguajes de programación (imperativa, orientado a objetos, funcionales, lógicos). Concurrencia y paralelismo. Uso de Heurísticas en algoritmos, paradigmas y lenguajes. Aplicaciones en laboratorio.

b) Programa analítico

Unidad 1: Traductores, Compiladores, Intérpretes.
Traductores: compiladores, ensambladores, preprocesadores, desensambladores. Intérpretes. Objetivos de la Traducción. Tipos de errores. Sistema de Procesamiento del Lenguaje. Editor de carga y enlace. Herramientas del entorno a la traducción. Introducción a la compilación. Fases de la compilación. Ventajas de la división en fases. Pasadas. Compiladores de compiladores.

Unidad 2: Introducción a los Lenguajes Formales
Definiciones de símbolo, alfabeto, cadena, cardinalidad o longitud de una cadena, cadena vacía y lenguaje formal. Operaciones sobre cadenas: concatenación, prefijo, sufijo, subcadena, potenciación. Operaciones sobre lenguajes: unión, concatenación, potenciación, clausura de Kleene, clausura positiva.

Unidad 3: Autómatas Finitos
Autómatas Finitos: Sistemas de estados finitos. Autómata Finito (AF): definición formal, comportamiento. Tablas de Transiciones. Diagrama de Transiciones. Función de transición extendida. Lenguaje reconocido por un AF. Estados muertos. Autómata Finito No Determinístico: Definición, diferencias y similitudes. AFN con λ-transiciones. Definición, diferencias y similitudes. Unión, intersección y complementación de AFs. Aplicación al análisis léxico.

Unidad 4: Expresiones Regulares.
Expresiones Regulares: Definición. Conjuntos regulares. Especificación de Lenguajes. Propiedades de las ER. Equivalencia de ERs y AFs: Transformación de una ER en un AFN. Transformación de un AF en una ER. Transformación de AFD a $AFD_{min.}$ Aplicación al análisis léxico.
Introducción a las Gramáticas. Introducción a las Máquinas de Turing. Jerarquía de Chomsky.

Unidad 5: Paradigmas de Programación y diseño de lenguajes
Introducción a los Paradigmas de programación: imperativo, orientado a objetos, funcional, lógico y concurrente. Conceptos fundamentales de cada paradigma.
Criterios de Diseño y de Implementación de Lenguajes de Programación.
Introducción a la programación funcional: Función. Transparencia Referencial. Efectos Laterales. Dominio. Recursividad. Funciones de Orden Superior. Funciones Anónimas. Introducción a LISP.

4- METODOLOGIA DE TRABAJO Y ESTRATEGIAS PEDAGOGICAS

La asignatura se dicta los días jueves y viernes, un módulo cada día.

Las clases se llevan a cabo mediante exposición dialogada, fomentando la participación de los alumnos.

Se utiliza software para la confección de autómatas y expresiones regulares.

Las técnicas principales de enseñanza-aprendizaje a ser utilizadas son:

Exposición Dialogada: para la presentación de los temas, con el uso de pizarrón y cañón y computadora para proyectar diapositivas y software de simulación/resolución de autómatas, expresiones regulares, gramáticas y máquinas de Turing. Además de exponer la teoría, se presentarán ejemplos y se estudiarán

casos especiales.

En algunos temas se utilizará la Problematización como medio de motivación y para darle sentido a la definición y explicación de los conceptos posteriores.

Se realizará Trabajo en Grupos, de 3 a 5 personas, para la resolución de ejercicios (guías de trabajos prácticos), que luego se exponen y corrigen utilizando el pizarrón. El trabajo grupal facilita la postura activa del alumno, que debe analizar el problema y posteriormente discutir las posibles soluciones con sus compañeros. Se recorrerán los grupos respondiendo consultas, motivando y ayudando en la resolución de los problemas.

Se seleccionarán algunos ejercicios para realizarlos en clase y los demás se deben resolver en horario extra-clase en software JFLAP o similar

Se utilizará la técnica de Proyecto para llevar a cabo un trabajo final sobre la definición de un lenguaje de programación y la construcción de un pequeño intérprete.

5- SISTEMA DE EVALUACION

Condiciones de regularidad y/o promoción

Se evalúa el trabajo en grupo y el individual, la actitud y la asistencia a clases.
- Evaluación grupal: se evalúan los trabajos prácticos que deben presentar.
-Evaluación individual:
Parcial 1: Autómatas
Parcial 2: Expresiones Regulares – Gramática
Evaluación en computadora de programación funcional
Existen instancias de recuperación para cada parcial, todas con posibilidad de promocionar

6- PROGRAMA DE TRABAJOS PRACTICOS

1. Autómatas finitos
2. Transformaciones
3. Expresiones Regulares
4. Programación funcional

7- BIBLIOGRAFIA

Bibliografía Principal:

Título: Introduction to Automata Theory, Languages, and Computation.
Autores: John Hopcroft, Rajeev Motwani, Jeffrey Ullman.
Editorial: Addison-Wesley.

Título: Compiladores. Principios, Técnicas y Herramientas.
Autores: Alfred Aho, Ravi Sethi, Jeffrey Ullman.
Editorial: Addison-Wesley Iberoamericana.

Título: Problem solving in Automata, Languages, and Complexity.
Autores: Ding-Zhu Du, Ker-I Ko.
Editorial: John Wiley & Sons.

Título: Programming Language Essentials.
Autores: Henri Bal. Dirk Grune.
Editorial: Addison-Wesley.

Bibliografía Complementaria:

Título: Formal Sintax and Semantics of Programming Languages.
Autores: Kenneth Slonneger, Barry L. Kurtz.
Editorial: Addison-Wesley.

Título: Desarrollo Modular de Procesadores de Lenguajes a partir de Especificaciones Semánticas Reutilizables.
Autor: Jose Emilio Labra Gayo – Tesis (2001).

8- REQUISITOS PARA RENDIR COMO ESTUDIANTES REGULARES, PROMOCIONALES Y LIBRES

Alumnos Regulares:
Objetivos alcanzados con notas entre 4 y 6. Debe rendir examen final

Alumnos Promocionados
Objetivos superados con notas entre 7 y 10. Promoción directa. No deben rendir final

Alumnos Libres
Objetivos no alcanzados, es decir notas inferiores a 4. Deben rendir final en condición de libres

9- CRONOGRAMA DE TRABAJO

SEMANA	TEMA		
1	**Introducción a la materia** **Evaluación Inicial**	Objetivos, relación con otras materias, contenido, bibliografía, evaluación y promoción, metodología. Conceptos básicos de programación, algoritmia y complejidad computacional	Exposición Trabajo grupal
2	**Traductores, Compiladores, Intérpretes.**	Traductores. Intérpretes. Objetivos de la Traducción. Tipos de errores. Sistema de Procesamiento del Lenguaje. Editor de carga y enlace. Herramientas del entorno a la traducción. Introducción a la compilación. Fases de la compilación. Ventajas de la división en fases. Pasadas. Compiladores de compiladores.	Exposición dialogada
3	**Introducción a los Lenguajes Formales**	Definiciones de símbolo, alfabeto, cadena, cardinalidad o longitud de una cadena, cadena vacía y lenguaje formal. Operaciones sobre cadenas: concatenación, prefijo, sufijo, subcadena, potenciación. Operaciones sobre lenguajes: unión, concatenación, potenciación, clausura de Kleene, clausura positiva. .	Exposición dialogada
4	**Autómatas Finitos**	Autómatas Finitos: Sistemas de estados finitos. Autómata Finito (AF): definición formal,	Exposición dialogada Trabajo

		comportamiento. Tablas de Transiciones. Diagrama de Transiciones. Función de transición extendida. Lenguaje reconocido por un AF. Estados muertos.	grupal
5	**Autómatas Finitos**	Autómata Finito No Determinístico: Definición, diferencias y similitudes. AFN con ⌜- transiciones. Definición, diferencias y similitudes. Unión, intersección y complementación de AFs. Aplicación al análisis léxico.	Exposición dialogada Trabajo grupal
6	**Expresiones Regulares**	Expresiones Regulares: Definición. Conjuntos regulares. Especificación de Lenguajes. Propiedades de las ER. Equivalencia de ERs y AFs: Transformación de una ER en un AFN-⌜	Exposición dialogada Trabajo grupal
7	**Expresiones Regulares**	Transformación de un AF en una ER. Aplicación al análisis léxico.	Exposición dialogada Trabajo grupal
8	**Evaluación Parcial I**		
9	**Autómatas finitos**	Transformación AFN-ε a AFD. Estados alcanzables. Funciones ε –clausura y Mueve	Exposición dialogada Trabajo grupal
10	**Autómatas finitos**	Transformación AFD a AFD_{min}	Exposición dialogada Trabajo grupal
11	**Gramáticas**	Definición. Gramáticas Independientes del Contexto.	Exposición dialogada Trabajo grupal
12	**Máquinas de Turing Jerarquía de Chomsky**	Definición de Máquina de Turing. Jerarquía de Chomsky	Exposición dialogada
13	**Diseño de Lenguajes de Programación**	Introducción a los Paradigmas de programación: imperativo, orientado a objetos, funcional, lógico y concurrente. Conceptos fundamentales de cada paradigma. Criterios de Diseño y de Implementación de Lenguajes de Programación. Entidades y ligaduras. Sistema de Tipos. Polimorfismo. Encapsulamiento y Abstracción.	Exposición dialogada Problematización Trabajo grupal
14	**Programación Funcional**	Introducción a la programación funcional: Función. Transparencia Referencial. Efectos Laterales. Dominio. Recursividad. Funciones de Orden Superior. Funciones Anónimas. Introducción a LISP.	
15	**Evaluación Parcial II**		
16	Proyecto	Presentación del proyecto y coloquios.	

10- FUNCIONES DE CADA INTEGRANTE DEL EQUIPO DE CATEDRA

La cátedra está conformada por:

Esp. Ing. Claudia Alvarez, docente a cargo de cátedra quien desarrolla las clases teóricas-prácticas y efectúa la corrección de las evaluaciones.

11- CRONOGRAMA DE ACTIVIDADES DE DOCENCIA, INVESTIGACION Y/O EXTENSION

12- MECANISMOS DE EVALUACION DE CATEDRA

Al comenzar el año lectivo, se realiza una evaluación inicial, para repasar conceptos referidos al área de programación.
Al finalizar el año lectivo, se realiza una evaluación crítica con los datos relevados del cursado de los alumnos, modificando y adaptando, sobre todo la didáctica de la enseñanza pues los alumnos actuales tienen otras inquietudes y necesitan de otra dinámica en cuanto a las actividades referidas a la enseñanza/aprendizaje