

1.- DATOS DE LA ASIGNATURA

Nombre de la asignatura: Teoría de la computación
Carrera: Ingeniería en Sistemas Computacionales
Clave de la asignatura: SCM - 0434
Horas teoría-horas práctica-créditos 3-2-8

2.- HISTORIA DEL PROGRAMA

Lugar y fecha de elaboración o revisión	Participantes	Observaciones (cambios y justificación)
Instituto Tecnológico de Toluca del 18 al 22 agosto 2003.	Representantes de la academia de sistemas y computación de los Institutos Tecnológicos.	Reunión nacional de evaluación curricular de la carrera de Ingeniería en Sistemas Computacionales.
Instituto Tecnológico de: Durango, Veracruz. 23 agosto al 7 de noviembre 2003.	Academia de sistemas y computación.	Análisis y enriquecimiento de las propuestas de los programas diseñados en la reunión nacional de evaluación.
Instituto Tecnológico de León 1 al 5 de marzo 2004.	Comité de consolidación de la carrera de Ingeniería en Sistemas Computacionales.	Definición de los programas de estudio de la carrera de Ingeniería en Sistemas Computacionales.

3.- UBICACIÓN DE LA ASIGNATURA

a). Relación con otras asignaturas del plan de estudio

Anteriores		Posteriores	
Asignaturas	Temas	Asignaturas	Temas
Estructura de datos.	Estructuras lineales y dinámicas.	Programación de sistemas.	Autómatas finitos determinísticos.
Programación Orientada a Objetos.	Se requiere el manejo de un lenguaje de alto nivel.	Sistemas digitales.	Lenguajes libres de contexto. Diseño de circuitos digitales utilizando máquinas de Moore y Mealy.
Matemáticas para computadoras.	Conjuntos Relaciones Teoría de grafos.	Sistemas operativos	Redes neuronales. Representación del conocimiento, Robótica

b). Aportación de la asignatura al perfil del egresado

Comprende la base teórica para la construcción de sistemas formales y utiliza técnicas de programación para modelarlos.

4.- OBJETIVO(S) GENERAL(ES) DEL CURSO

El estudiante comprenderá la base teórica para la construcción de sistemas formales y utilizará técnicas de programación para modelarlos.

5.- TEMARIO

Unidad	Temas	Subtemas
1	Introducción.	1.1 Autómatas, computabilidad y complejidad. 1.2 Nociones matemáticas. 1.2.1 Conjuntos 1.2.2 Funciones y Relaciones 1.2.3 Cadenas y Lenguajes 1.3 Inducción matemática.
2	Lenguajes regulares.	2.1 Autómatas finitos 2.1.1 Autómatas finitos determinísticos. 2.1.2 Autómatas finitos No determinísticos 2.2 Expresiones regulares. 2.3 Lenguajes no regulares.
3	Lenguajes libres de contexto.	3.1 Gramáticas libres de contexto. 3.2 Árboles de derivación. 3.3 Formas normales de Chomsky. 3.4 Formas normales de Greibach. 3.5 Eliminación de Factores Comunes izquierdos. 3.6 Eliminación de recursividad izquierda. 3.7 Eliminación de la ambigüedad. 3.8 Autómatas Push-Down. 3.9 Lenguajes no regulares.
4	Máquina de Turing.	4.1 Definición formal de una máquina de Turing. 4.2 Construcción modular de una máquina de Turing. 4.3 Lenguajes aceptados por la MT. 4.4 Variantes de una máquina de Turing. 4.5 Problemas de Hilbert.
5	Decidibilidad.	5.1 Lenguajes Decidibles. 5.2 El problemas de Halting. 5.3 Decidibilidad de Teorías Lógicas.

5.- TEMARIO (Continuación)

6	Reducibilidad.	6.1 Problemas insolubles para la teoría de lenguajes. 6.2 Un problema simple insoluble. 6.3 Funciones computables. 6.4 Reducibilidad de Turing.
---	----------------	--

6.- APRENDIZAJES REQUERIDOS

- Conocer la teoría vista en matemáticas discretas, como base conjuntos, funciones y relaciones.
- Conocer y manejar las estructuras de datos, su representación y programación.
- Conocer y manejar lenguajes de programación de alto nivel.

7.- SUGERENCIAS DIDÁCTICAS

- Investigación previa a la clase de los conceptos de la asignatura, por equipos analizarlos y discutirlos
- Propiciar el trabajo por equipos, exposición, discusión grupal, entre otros
- Plantear y analizar casos típicos
- Desarrollar prácticas en laboratorio para modelar casos tipo
- Realizar ejercicios como reforzamiento de temas
- Realizar dinámicas de grupo que permitan reforzar la teoría

8.- SUGERENCIAS DE EVALUACIÓN

- Evaluación teórica
- Elaboración de ejercicios
- Prácticas de laboratorio para modelar a través de lenguajes computacionales
- Prácticas en laboratorio de electrónica para la programación de PLC's o utilizar un simulador
- Visitas a laboratorios de Ingeniería Industrial para conocer el funcionamiento de un CIM o a través de un simulador
- Trabajos de investigación (artículos, libros, Internet, etc.)
- Elaboración de ensayos y artículos sobre Teoría de la Computación

9.- UNIDADES DE APRENDIZAJE

UNIDAD 1.- Introducción.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
El estudiante reafirmará las bases matemáticas necesarias para la teoría de la computación.	<ol style="list-style-type: none">1.1 Realizar ejercicios de conjuntos, funciones y relaciones.1.2 Realizar análisis de grafos.1.3 Realizar análisis de complejidad en algoritmos.1.4 Realizar ejercicios en donde se aplique Inducción matemática.1.5 Analizar la complejidad de algoritmos y realizar modificaciones que mejoren su desempeño.1.6 Investigar acerca de la teoría de la computación, las bases que lo soportan así como sus aplicaciones.	

UNIDAD 2.- Lenguajes regulares.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
Representará lenguajes a través de autómatas, expresiones regulares y su aplicación.	<ol style="list-style-type: none">2.1 Desarrollar ejercicios para la representación de lenguajes por medio de AFD, AFN, AFN-ϵ y expresiones regulares.2.2 Utilizar un lenguaje de programación de alto nivel para representar expresiones regulares.2.3 Realizar prácticas de laboratorio para la programación de PLC's, como casos de aplicación de autómatas.2.4 Desarrollar una herramienta que genere código libre de errores a partir de la representación gráfica de autómatas.2.5 Investigar que otras aplicaciones tiene la teoría de lenguajes regulares.	

UNIDAD 3.- Lenguajes libres de contexto.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
Comprenderá la teoría de lenguajes de contexto libre y su representación.	<ul style="list-style-type: none">3.1 Identificar los diferentes tipos de lenguajes de acuerdo a la clasificación de Chomsky.3.2 Realizar ejercicios que permitan desarrollar la habilidad para representar lenguajes libres de contexto.3.3 Utilizar un lenguaje de alto nivel para representar lenguajes libres de contexto, solamente como casos tipo.3.4 Investigar otros usos que se le puede dar a la teoría de lenguajes libres de contexto.3.5 Investigar nuevas técnicas para la representación de lenguajes libres de contexto.	

UNIDAD 4.- Máquina de Turing.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
Comprenderá la representación de lenguajes y funciones en una máquina de Turing.	<ul style="list-style-type: none">4.1 Realizar ejercicios que permitan la representación de operaciones matemáticas básicas como suma, resta, multiplicación, potencia, entre otros.4.2 Utilizar la teoría para la representación de lenguajes.4.3 Simular a través de un lenguaje de alto nivel, la representación de una máquina de Turing.	

UNIDAD 5.- Decibilidad.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
Comprenderá la teoría de la decibilidad aplicada a lenguajes.	<p>5.1 Desarrollar problemas de decibilidad aplicado a lenguajes regulares.</p> <p>5.2 Desarrollar problemas de decibilidad aplicado a lenguajes libres de contexto.</p> <p>5.3 Analizar problemas en donde se aplique la decibilidad.</p> <p>5.4 Investigar otras áreas del conocimiento en donde se aplique la teoría de decibilidad.</p> <p>5.5 Desarrollar un ensayo a partir de los resultados de la investigación realizada en el punto 5.4.</p> <p>5.6 Desarrollar, a través de un lenguaje de alto nivel, problemas tipo de decibilidad.</p> <p>5.7 Investigar el teorema de Godel.</p>	

UNIDAD 6.- Reducibilidad.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
Aplicará la teoría de la reducibilidad.	<p>6.1 Resolver problemas de undecibilidad en la teoría lenguajes.</p> <p>6.2 Analizar casos en donde se requiera la aplicación de la reducibilidad.</p> <p>6.3 Elaborar un ensayo a partir de los casos analizados en el punto 6.2.</p> <p>6.4 Desarrollar a través de un lenguaje de alto nivel, problemas tipo de reducibilidad.</p>	

10. FUENTES DE INFORMACIÓN

1. Martin, John C.
Introduction to Languages and the Theory of Computation.
Ed. Prentice Hall.
2. Sipser, Michael.
Introduction to the Theory of Computation.
Ed. PWS Publishing Company.
3. Cohen, Daniel I.A.
Introduction to Computer Theory.
Ed. Wiley.
4. Davis, Martín D., Weyuker, Elaine.
Computability, Complexity and Languages Fundamentales of Theoretical Computer Science.
Ed. Academic Press.
5. Denning, Peter J.
Machines, Languages and Computation.
Ed. Prentice Hall.
6. Hopcroft, John, Ullman, Jeffrey.
Introduction to Automatas Theory, Languages and Computation.
Ed. Addison-Wesley.
7. Kelley, Dean.
Teoría de Automatas y Lenguajes Formales.
Ed. Prentice Hall.
8. Lewis, Larry., Papadimitriou, Christos H.
Elements of the Theory of Computation.
Ed. Prentice Hall.
9. Rayward-Smith, V.S.
A First Course in a Formal Language Theory.
Ed. Mc Graw Hill.
10. Jeffrey E.F. Friedl.
Mastering Regular Expressions.
Ed. O'reilly & Associates, Inc.
11. Brookshear.
Teoría de la Computación, Lenguajes Formales, Autómatas y Complejidad.
Ed. Addison Wesley.

12. Isasi, Martínez y Borrajo.
Lenguajes, Gramáticas y Autómatas.
Ed. Addison Wesley.

11. PRÁCTICAS

Unidad Práctica

- 1 Analizar la complejidad de un algoritmos y modificarlo para mejorar su desempeño. Los casos propuestos, podrán estar relacionados con métodos de ordenamiento iterativos o recursivos
- 2 Desarrollar a través de un lenguaje de programación de alto nivel la representación de lenguajes simples a través de AFD's.
- 3 Realizar prácticas en el laboratorio de electrónica para la programación de PLC's, como casos de aplicación de autómatas o en su defecto el uso de simuladores de software.
- 4 Desarrollar una herramienta de software que genere código libre de errores a partir de la representación gráfica de autómatas.
- 5 Analizar la funcionalidad de flex o lex como herramientas orientadas a la generación de código para compilador de compilador.
- 6 Utilizar un lenguaje de alto nivel para representar lenguajes libres de contexto, solamente como casos tipo.
- 7 Analizar la funcionalidad de yacc o bison, como herramientas orientadas a la generación de código para compilador de compilador.
- 8 Simular a través de un lenguaje de alto nivel, la representación de una máquina de Turing