UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERIA
Escuela de Ciencias y Sistemas

FICHA TÉCNICA DEL CURSO: Lenguajes Formales y de Programación

<table>
<tr><td>No.</td><td colspan="3">Descripción</td></tr>
<tr><td>.</td><td>Código 796</td><td colspan="2">Créditos 3</td></tr>
<tr><td>1</td><td>Escuela
Ciencias y Sistemas
https://dtt-ecys.org</td><td>Área a la que pertenece
Ciencias de la computación</td><td>Vigencia
Primer Semestre 2019</td></tr>
<tr><td>2</td><td>Horas por semana
2</td><td colspan="2">Horario
Martes 7:00 a 8:40</td></tr>
<tr><td>3</td><td colspan="3">Pre-requisitos:
770 Introducción a la Programación 1
795 Lógica de sistemas
960 Matemática de Cómputo 1</td></tr>
<tr><td>4</td><td colspan="3">Postrequistios:
777 Organización de lenguajes y compiladores 1
772 Estructuras de datos</td></tr>
<tr><td>5</td><td colspan="3">Secciones: A-, B-</td></tr>
<tr><td>6</td><td colspan="3">I. Descripción General

Este curso busca introducir al estudiante con los fundamentos teóricos matemáticos y conceptos que fundamentan los lenguajes de programación. El estudiante debe adquirir la base teórica necesaria y requerida para que pueda llevar un curso avanzado de lenguajes y compiladores.
Se busca, además, definir los modelos matemáticos asociados a la representación de los diferentes tipos de lenguajes para luego implementar estos conceptos en lenguajes de programación.
Es de primordial importancia que pueda reconocer cualquier tipo de gramática, pero sobre todo, pueda manejar y diseñar gramáticas para lenguajes regulares y para lenguajes libres de contexto, además, de los modelos matemáticos que las resuelven. Se busca que el estudiante tenga mucha práctica en el diseño de gramáticas para representar lenguajes y que adquiera la habilidad de diseñarlas sin problema. Adquiriendo conceptos y los pueda relacionar a los aspectos técnicos y prácticos conociendo su aplicación en lenguajes reales conocidos. El estudiante debe aprender la teoría que esta atrás de los diferentes componentes de un compilador, las técnicas de programación usadas para poner esta teoría en practica,
Al finalizar el curso el estudiante estará en la capacidad de poder recibir un curso avanzado de compiladores.

II. Objetivos
Objetivo General
Que el estudiante tenga los conceptos teóricos y matemáticos necesarios, que fundamentan los lenguajes de programación y el diseño de lenguajes y compiladores.
Objetivos Específicos

Al final del curso el estudiante deberá:
1. Definir cualquier lenguaje formal
2. Reconocer las características que identifican a cualquier tipo de gramática.
3. Manejar la terminología de los lenguajes y compiladores.
4. Conocer el modelo matemático que resuelve cada tipo de gramática.
5. Diseñar gramáticas que representen lenguajes específicos
6. Conocer el funcionamiento de un analizador léxico y su implementación
7. Conocer e implementar maquinas de estado finito
8. Diseñar e implementar gramáticas libres de contexto
9. Conocer los conceptos que fundamentan el análisis sintáctico
10. Aplicar los modelos matemáticos que resuelven cualquier gramática</td></tr>
</table>

III. Contenido

Contenido	Planificación
Unidad 1. Lenguajes Formales 1. Definiciones Lenguajes de programación, Compiladores e interpretes Generaciones de lenguajes Partes del compilador: Análisis sintáctico Análisis léxico Definición de tokens, lexemas, palabras reservadas. 2. Lenguajes formales 2.1 Definición 2.2 Símbolos terminales 2.3 Símbolos no terminales 2.4 Gramática 2.5 Estado inicial	• 3 períodos de clase
Unidad 2 : Jerarquía de Chomsky 1. Jerarquía de Chomsky 1.1 Lenguajes Recursivamente enumerables Máquinas de Turing 1.2 Lenguajes Sensibles al Contexto Autómatas lineales limitados 1.3 Lenguajes Libres de contexto Autómatas Descendentes 1.4 Lenguajes regulares Autómatas finitos	• 3 períodos de clase • ***Primera evaluación (12/Febrero)***
Unidad 3: Lenguajes regulares 1. Lenguajes regulares 2. Gramáticas Regulares (tipo 3) 3. Diseño de gramáticas regulares 3.1 Ejemplos y ejercicios de gramáticas regulares 3.2 Aplicación de gramáticas regulares 4. Expresiones regulares 5. Autómatas Finitos 5.1 Grafos para representación de autómatas 5.2 Autómatas Finitos No Determinísticos NFA 5.3 Autómatas Finitos Determinísticos DFA 6. Método para pasar de gramáticas a expresiones regulares y de expresiones regulares a gramáticas 7. Métodos para Calcular DFAs 7.1 Construcción de Thomson y minimización de estados 7.2 Método del árbol 8. Ejemplos y ejercicios	• 8 períodos de clase • ***Segunda evaluación (19/Marzo)***
Unidad 4: Lenguajes libres de contexto 1. Lenguajes Libres de contexto 2. Gramáticas Libres de contexto (Tipo 2) 3. Diseño de gramáticas libres de contexto 3.1 Ejemplos y ejercicios de gramáticas libres de contexto 3.2 Aplicación de lenguajes libres de contexto 4. Recursividad por la izquierda y recursividad por la derecha 5. Gramáticas ambiguas 6. Parser recursivos descendentes 7. Ejemplos y ejercicios	• 6 períodos de clase • ***Tercera evaluación (30/abril)***

IV. Metodología:

El curso se desarrollará intercalando clases magistrales para la exposición de conceptos nuevos y clases participativas, en las que se asume que el estudiante realiza las lecturas, tareas o ejercicios dejados para realizar fuera de clase, previo al inicio de un nuevo día de clase.

V. Evaluación:

La nota final estará compuesta de 100 puntos, distribuidos de la siguiente manera:

3 Evaluaciones de rendimiento (15 puntos c/u).....	45 puntos
Tareas, trabajos en clase, comprobaciones, asistencia, cortos, etc.	10 puntos
Laboratorio (proyectos, prácticas, etc.).....	20 puntos
Evaluación Final.....	25 puntos

Nota Total.....	100 puntos

VI. Observaciones:

- Será necesario contar con un 80% de asistencia y ***aprobar el laboratorio del curso con una nota mínima de 61 puntos***, para tener derecho a la evaluación final.
- Se realizaran exámenes cortos cada día de clase y se debe tener un 80% de exámenes realizados, para tener derecho a la evaluación final.
- En este curso, no se pasan notas de semestres anteriores, no se guardan notas para semestres posteriores, y no se aceptan estudiantes con problemas de prerrequisitos.

7	Bibliografía	1. Aho, Alfred V., Sethi y Ullman. **Compiladores: principios, técnicas y herramientas.** Addison-Wesley. 2. Brookshear, J. Glenn. **Teoría de la Computación** - Lenguajes formales, autómatas y complejidad. Addison-Wesley Iberoamericana. 3. Andrew W. Appel. **Modern Compiler Implementation in Java**. Second Edition. Cambridge University Press. 4. Hopcroft, John. y Ullman, Jeffrey. **Introducción a la Teoría de Autómatas, Lenguajes y Computación.**
8	No. De Secciones	4
9	Catedráticos titulares y auxiliares	Sección A- Inga. Damaris Campos de López lfyp.usac@gmail.com Sección B- Inga. Zulma Aguirre aguirrezulma@gmail.com
10	Director de Escuela	**Ing. Marlon Pérez Türk**