UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERIA
ESCUELA DE SISTEMAS

NOMBRE DEL CURSO: Laboratorio de Lenguajes Formales y de Programación

CÓDIGO:	796	**CRÉDITOS:**	3
ESCUELA:	Ciencias y Sistemas	**ÁREA A LA QUE PERTENECE:**	Ciencias de la computación
PRERREQUISITOS:	770 – Introducción a la Programación 1 795 – Lógica de sistemas 960 – Matemática de cómputo 1	**POST REQUISITOS:**	777 Organización de Lenguajes y Compiladores 1 772 Estructuras de Datos
CATEGORÍA:	Obligatorio	**SEMESTRE:**	2do Semestre. 2021
CATEDRÁTICO:	Ing. Otto Amilcar Rodriguez Acosta	**AUXILIAR:**	Fernando Feliciano Chajon del Cid
EDIFICIO:	Virtual	**SECCIÓN:**	A+
SALÓN DEL CURSO:	Salón 22 – Google Meet	**SALÓN DEL LABORATORIO:**	Salón 25 – Google Meet
HORAS POR SEMANA DEL CURSO:	2	**HORAS POR SEMANA DEL LABORATORIO:**	2
DÍAS QUE SE IMPARTE EL CURSO:	Martes	**DÍAS QUE SE IMPARTE EL LABORATORIO:**	Sábado
HORARIO DEL CURSO:	07:10 – 08:50 HRS	**HORARIO DEL LABORATORIO:**	13:50 – 15:30 HRS

DESCRIPCIÓN DEL CURSO:

El laboratorio tiene como propósito introducir al estudiante de ciencias de la computación al estudio, análisis, comprensión e implementación de lenguajes de programación bajo una estructura genérica que contribuya al desarrollo de un compilador básico y funcional; abarcando las fases de análisis léxico, análisis sintáctico e introducción al análisis semántico.

Comprende también mostrar métodos que faciliten la creación de analizadores léxicos por medio de algoritmos programados.

OBJETIVOS:

Objetivo General

- Introducir al estudiante al conocimiento y desarrollo de las funciones básicas de los compiladores, y los ponga en práctica en la construcción de las primeras fases de esté.

Objetivos Específicos

- Aplicar los conocimientos adquiridos en clase para implementar analizadores léxicos.
- Implementar un analizador sintáctico utilizando las técnicas vistas en clase.

METODOLOGIA:

- Se imparten clases presenciales con material de apoyo que es proporcionado al estudiante al finalizar la clase.
- En el transcurso del semestre se realizarán tareas y ejercicios prácticos para evaluar los conocimientos adquiridos por el estudiante.
- Se realizan prácticas y proyectos en donde se ponga en práctica y se puedan evaluar los conceptos adquiridos en el curso, tomando en cuenta que pueden incluirse temas de cursos prerrequisito.
- Se impartirán clases prácticas donde se resuelvan problemas relacionados con el tema a desarrollar.

REQUISITOS:

- El laboratorio se debe aprobar con nota mínima de 61 puntos.
- Solo se calificarán exámenes, proyectos y demás actividades de estudiantes asignados en el curso.
- 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.
- Copias parciales o totales en las tareas, investigaciones, etc. serán sancionadas con una nota de cero.
- Copias en los proyectos y prácticas serán sancionadas con una nota de cero y reportadas a la Escuela de Sistemas.
- Las tareas, investigaciones, prácticas, proyectos deben ser entregadas en la fecha indicada y con el formato establecido.

EVALUACION:

Aspecto:	**Valor:**
Tareas, asistencia, participación y evaluaciones	*10 pts.*
Practicas	
Práctica 1: Publicación: 29/07/2021 Entrega: 19/08/2021	*15 pts.*
Proyectos	
Proyecto 1: Publicación: 20/08/2021 Entrega: 23/09/2021	*30 pts.*
Proyecto 2: Publicación: 24/09/2021 Entrega: 28/10/2021	*35 pts.*
Examen Final	*10 pts.*
Total	***100 pts.***

CALIFICACIÓN DE PRÁCTICAS Y PROYECTOS

- La calificación de las prácticas y proyectos se realizarán presencialmente y desde los archivos ejecutables entregados.
- No se puede agregar o quitar algún símbolo en los archivos de entrada. La práctica o el proyecto deberá funcionar con los archivos que sean proporcionados por el auxiliar para la calificación.
- Existirán horarios para la calificación, por lo cual el estudiante deberá de elegir el horario que mejor le convenga.
- Anomalías o copias detectadas de entregables tendrán de manera automática una nota de 0 puntos y los involucrados serán reportados a la Escuela de Ingeniería en Ciencias y Sistemas, para que se apliquen las sanciones correspondientes.

1. Introducción

1.1. Procesadores de lenguaje

1.2. La estructura de un compilador

1.2.1. Análisis léxico

1.2.2. Análisis sintáctico

1.2.3. Análisis semántico

1.2.4. Generación de código intermedio

1.2.5. Optimización de código

1.2.6. Generación de código

1.2.7. Administración de la tabla de símbolos

1.2.8. El agrupamiento de fases en pasadas

1.2.9. Herramientas de construcción de compiladores

1.3. La evolución de los lenguajes de programación

1.3.1. El avance a los lenguajes de alto nivel

1.3.2. Impactos en el compilador

1.4. La ciencia de construir un compilador

1.4.1. Modelado en el diseño e implementación de compiladores

1.4.2. Lenguajes formales

1.4.3. Definiciones

1.4.4. Jerarquía de Chomsky

2. Análisis léxico

2.1. La función del analizador léxico

2.1.1. Comparación entre análisis léxico y análisis sintáctico

2.1.2. Tokens, patrones y lexemas

2.1.3. Atributos para los tokens

2.1.4. Errores léxicos

2.2. Uso de búfer en la entrada

2.2.1. Pares de búferes

2.2.2. Centinelas

2.3. Especificación de los tokens

2.3.1. Cadenas y lenguajes

2.3.2. Operaciones en los lenguajes

2.3.3. Definiciones regulares (Gramáticas regulares)

2.3.4. Expresiones regulares

2.3.5. Extensiones de las expresiones regulares

2.4. Reconocimiento de tokens

2.4.1. Diagrama de transición de estados

2.4.2. Reconocimiento de las palabras reservadas y los identificadores

2.4.3. Finalización del bosquejo

2.4.4. Arquitectura de un analizador léxico basados en diagramas

2.5. Autómatas finitos

2.5.1. Autómatas finitos no deterministas

2.5.2. Tablas de transición

2.5.3. Aceptación de las cadenas de entrada mediante los autómatas

2.5.4. Autómatas finitos deterministas

2.6. De las expresiones regulares a los autómatas

2.6.1. Conversión de un AFN a AFD

2.6.2. Simulación de un AFN

2.6.3. Eficiencia de la simulación de un AFN

2.6.4. Construcción de una AFN a partir de una expresión regular

2.6.5. Eficiencia de los algoritmos de procesamiento de cadenas

2.7. Diseño de un generador de analizadores léxicos

2.7.1. La estructura del analizador generado

2.7.2. Coincidencia de patrones con base en los AFNs

2.7.3. AFD para analizadores léxicos
2.7.4. Implementación del operador de preanálisis
2.8. Optimización de los buscadores por concordancia de patrones basados enAFD
2.8.1. Estados significativos de una AFN
2.8.2. Funciones calculadas a partir del árbol sintáctico
2.8.3. Cálculo de anulable, primera posición y última posición
2.8.4. Cálculo de siguiente posición
2.8.5. Conversión directa de una expresión regular a un AFD
2.8.6. Minimización del número de estados de un AFD
2.8.7. Minimización de estados en los analizadores léxicos
2.8.8. Intercambio de tiempo por espacio en la simulación de un AFD

3. Introducción a la sintaxis

3.1. Introducción
3.2. Definición de sintaxis
3.2.1. Definición de gramáticas
3.2.2. Derivaciones
3.2.3. Árboles de análisis sintáctico
3.2.4. Ambigüedad
3.2.5. Asociatividad de los operadores
3.2.6. Precedencia de los operadores
3.3. Traducción orientada a la sintaxis
3.3.1. Notación prefija
3.3.2. Atributos sintetizados
3.3.3. Definiciones simples orientadas a la sintaxis
3.3.4. Recorrido de árboles
3.3.5. Esquemas de traducción

4. Análisis sintáctico

4.1.1. Análisis sintáctico tipo arriba-abajo
4.1.2. Análisis sintáctico predictivo
4.1.3. Cuando usar las producciones épsilon
4.1.4. Diseño de un analizador sintáctico predictivo
4.1.5. Recursividad por la izquierda
4.2. Un traductor para las expresiones simples
4.2.1. Sintaxis abstracta y concreta
4.2.2. Adaptación del esquema de traducción
4.2.3. Procedimientos para los no terminales
4.2.4. Simplificación del traductor
4.2.5. El programa completo
4.3. Tabla de símbolos
4.3.1. Tabla de símbolos por alcance
4.3.2. El uso de las tablas de símbolos

BIBLIOGRAFÍA:

- Aho, Alfred V., Sethi y Ullman. Compiladores: principios, técnicas y herramientas. Addison Wesley.
- Brookshear, J. Glenn. Teoría de la Computación - Lenguajes formales, autómatas y complejidad. AddisonWesley Iberoamericana.

John E Hopcroft. introducción a la Teoría de Autómatas, Lenguajes y computación.