UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS

PROGRAMA DE LABORATORIO

CÓDIGO:	777	**CRÉDITOS:**	4
NOMBRE CURSO:	Organización de Lenguajes y Compiladores 1	**SECCIÓN:**	A
ESCUELA:	Ciencias y Sistemas	**ÁREA A LA QUE PERTENECE:**	Ciencias de la Computación
PRE-REQUISITO:	771 – Introducción a la Programación y Computación 796 – Lenguajes Formales y de Programación 962 – Matemática para Computación 2	**POS-REQUISITO:**	781 – Organización de Lenguajes y Compiladores 2 2036 – Practicas Intermedias
CATEGORÍA:	Obligatorio	**SEMESTRE:**	Primer Semestre 2019
CATEDRÁTICO (A):	Ing. Mario José Bautista Fuentes	**AUXILIAR:**	Nery Galvez
EDIFICIO:	T-3	**EDIFICIO:**	T-3
SALÓN DEL CURSO:	113	**SALÓN DE LABORATORIO:**	309
HORAS POR SEMANA DEL CURSO:	4	**HORAS POR SEMANA DEL LABORATORIO:**	2
DÍAS QUE SE IMPARTE EL CURSO:	Sábado	**DÍAS QUE SE IMPARTE EL LABORATORIO:**	Lunes
HORARIO DEL CURSO:	10:30 AM – 13:50 PM	**HORARIO DEL LABORATORIO:**	10:50 AM – 12:30 PM

DESCRIPCIÓN DEL LABORATORIO:

El laboratorio del curso de Organización de Lenguajes y Compiladores 1, trata sobre la parte práctica del curso con la aplicación de las primeras fases del compilador, la fase de análisis que incluye lo que es análisis lexicográfico, análisis sintáctico y la introducción al análisis semántico, también se incluye el manejo de errores y de la tabla de símbolos.

OBJETIVO GENERAL:

Poner en práctica los elementos que conforman el análisis y la síntesis en el proceso de compilación con el uso de los diferentes tipos de herramientas de compilación

OBJETIVOS ESPECÍFICOS:

1. Que el estudiante aprenda en detalle las primeras fases del proceso de compilación, principalmente el análisis lexicográfico y el análisis sintáctico.
2. Aprender a desarrollar aplicaciones que compilen una entrada determinada y ejecutar dicho código para obtener salidas del programa hacia el usuario.
3. Crear proyectos de innovación descubriendo detrás de los mismos el proceso de compilación de un programa.
4. Fomentar en el estudiante el análisis en la resolución de problemas complejos en la lectura de entradas determinadas para un compilador.

HABILIDADES:

1. Comprensión de la estructura y funciones de un compilador.
2. Compresión de las estructuras de datos utilizadas en el proceso de compilación y su relación con las distintas fases del compilador.
3. Comprensión de la importancia y el manejo de errores en la construcción se software.
4. Comprensión de la relación entre hardware y software durante la ejecución de un programa computacional.

COMPETENCIAS:

1. Dominar los conceptos básicos de compiladores.
2. Elegir con criterio la manera óptima de codificar una solución informática.
3. Aplicar herramientas de análisis léxico y sintáctico para la resolución de problemas.
4. Crear soluciones funcionales aplicando los conceptos de compiladores.

METODOLOGÍA:

1. Clases para la explicación de conceptos, resolución de ejercicios prácticos y resolución de dudas.
2. Elaboración de tareas, hojas de trabajo, exámenes cortos, practicas, proyectos

EVALUACIÓN DEL RENDIMIENTO ACADÉMICO: el laboratorio tiene una ponderación de 40 sobre la nota final del curso, distribuido de la siguiente manera

3 Hojas de Trabajo	1% c/u	3%
2 Exámenes cortos	2.5% c/u	5%
Tareas, Investigaciones, etc.		5%
2 Practicas	10% c/u	20%
Proyecto 1		22%
Proyecto 2		35%
Examen Final		10%
	TOTAL	**100%**

OBSERVACIONES:

1. La calificación de prácticas y proyectos de laboratorio es personal acoplándose al día y horario que se indique previamente.
2. Copias parciales o totales de prácticas o proyectos tendrán una nota de 0 puntos y los responsables serán reportados a la Escuela de Ciencias y Sistemas.
3. Todas las tareas e investigaciones que sean identificadas como copias parciales o tendrán una nota de 0 puntos.
4. Para aprobar el laboratorio se debe tener una nota final igual o mayor a 61 puntos.

CONTENIDO:

1. INTRODUCCIÓN A LA COMPILACIÓN

1.1. Conceptos generales de compiladores
- 1.1.1. Definición de compilador
- 1.1.2. Definición de interprete
- 1.1.3. Diferencias entre compilador e interprete

1.2. Tipos de Compiladores
1.3. Fases de Análisis
1.4. Fases de Síntesis

2. ANÁLISIS LÉXICO Y SINTÁCTICO

2.1 ANÁLISIS LÉXICO
- 2.1.1. Definición.
- 2.1.2. Componentes: Token, Lexema, Patrón (Expresiones Regulares)
- 2.1.3. Autómatas (Método del Árbol, método de Thompson)
- 2.1.4. Manejo de errores léxicos

2.2 ANÁLISIS LÉXICO
- 2.2.1 Definición
- 2.2.3 Gramáticas independientes del contexto (Libres del contexto)
- 2.2.4 Derivación
- 2.2.4 Árbol de Análisis Sintáctico

3. MODULO HERRAMIENTA JFLEX Y CUP

3.1 Instalación
3.2 Definición de expresiones y Gramática,
3.3 Implementación de archivos .flex y .cup
3.4 Descripción de las clases generadas.
3.5 Implementación de acciones.
3.6 Manejo de Errores.

4. GRAMÁTICAS INDEPENDIENTES DEL CONTEXTO

4.1 Definición
4.2 Funciones Primero y Siguiente
4.3 GRAMÁTICAS DESCENDENTES
- 4.3.1 Definición
- 4.3.2 Algoritmos de familia LL($_k$)
- 4.3.3 Ejemplos

4.4 Re-escritura gramaticales
- 4.4.1 Supresión de ambigüedad
- 4.4.2 Factorización

4.4.3 Eliminación de recursividad por la izquierda

4.5 GRAMÁTICAS ASCENDENTES
- 4.5.1 Definición
- 4.5.2 Algoritmos de familia LR($_k$)
- 4.5.3 Ejemplos

5. MODULO HERRAMIENTAS FLEX/BISON

5.1 FLEX
- 5.1.1 de usuario
- 5.1.2 Directivas del analizador
- 5.1.3 Declaraciones y definiciones de Expresiones regulares y estados
- 5.1.4 Reglas léxicas

5.2 BISON
- 5.2.1 Código de usuario
- 5.2.2 Declaraciones, definiciones símbolos gramaticales, precedencia y asociativo
- 5.2.3 Definición de gramáticas
- 5.2.4 Manejo de errores

6. MODULO HERRAMIENTA IRONY

6.1 Descarga, Instalación, Compilación
6.2 Sintaxis de escritura de expresiones regulares y gramáticas
6.3 Comunicación con el entorno de programación
6.4 Acciones gramaticales
6.5 Manejo de Errores

7. TABLA DE SÍMBOLOS Y MANEJO DE ERRORES

7.1 Definición
7.2 Estructura
7.3 Operaciones con la tabla de símbolos
7.4 Técnicas de manejo y recuperación de errores

8. REPRESENTACIÓN DE CÓDIGO DE UNA GRAMÁTICA

8.1 Traducción dirigida por la sintaxis
- 8.1.1 Definición dirigida por la sintaxis
- 8.1.2 Atributos sintetizados
- 8.1.3 Atributos heredados

8.2 Construcción de Arboles de sintaxis abstracta (AST)
8.3 Ejecución de código

CALENDARIZACIÓN DE ACTIVIDADES:

1. Primera práctica:
   1.1. **Publicación de enunciado:** Jueves 7 de febrero.
   1.2. **Explicación:** Lunes 11 de Febrero.
   1.3. **Entrega:** Viernes 22 de Febrero.

2. Primer Proyecto:
   2.1. **Publicación de enunciado:** Viernes 22 de Febrero
   2.2. **Explicación:** Lunes 25 de Febrero
   2.3. **Entrega:** Jueves 21 de Marzo

3. Segunda práctica:
   3.1. **Publicación de enunciado:** Jueves 21 de Marzo
   3.2. **Explicación:** Lunes 25 de Marzo
   3.3. **Entrega:** Jueves 4 de Abril

4. Segundo Proyecto:
   4.1. **Publicación de enunciado:** Jueves 4 de Abril
   4.2. **Explicación:** Lunes 8 de Abril
   4.3. **Entrega:** Jueves 2 de Mayo

5. Examen Final:
   5.1. Viernes 3 de Mayo

6. Conferencia:
   6.1. La fecha de dará en el trascurso del laboratorio

7. Tareas, Hojas de Trabajo, Exámenes Cortos de laboratorio:
   7.1. Las fechas se darán en el transcurso del laboratorio

BIBLIOGRAFÍA:

1. Principios, Técnicas y Herramientas Aho, Sethi y Ullmam. PEARSON ADDISON- WESLEY, 2008, 2da. Edición