

FICHA TÉCNICA DEL CURSO: ORGANIZACIÓN DE LENGUAJES Y COMPILADORES 1

No.	Descripción		
	Código 777	Créditos 4	
1	Escuela Ciencias y Sistemas	Área a la que pertenece Ciencias de la Computación	Vigencia Primer Semestre 2020
2	Horas por semana 4 horas	Horario Secciones B y C; Martes y Jueves 07:00 – 09:00 Sección A: Sábado 10:30 – 13:50	
3	Prerrequisitos: 771 (Introducción a la Programación y Computación 2) 796 (Lenguajes Formales y de Programación) 962 (Matemática de Computo 2)		
4	Post-requisitos: 781 (Organización de Lenguajes y Compiladores 2)		
5	Sección: A, B y C		
6	<p>I. Descripción General                      Este curso estudia los principios básicos de un compilador y / o intérprete, partiendo de la estructura interna del proceso de compilación, y describiendo las fases de este proceso.                      Se tratan en detalle las primeras fases del proceso: análisis lexicográfico, análisis sintáctico y traducción dirigida por la sintaxis. Para poner en práctica los conceptos aprendidos se realizan varias tareas y proyectos prácticos.</p> <p>II. Objetivos  <i>Objetivo General</i>                      Que el estudiante adquiriera una base teórica fundamental para el entendimiento de la estructura interna del proceso de compilación.  <i>Objetivos Específicos</i></p> <ul style="list-style-type: none"> <li>• Que el estudiante aprenda con detalle las primeras fases del proceso de compilación, principalmente el análisis lexicográfico y el análisis sintáctico.</li> <li>• Capacitar y ejercitar al estudiante en los principios del análisis, diseño e implementación de compiladores, para lo cual se realizarán varias tareas y proyectos</li> </ul> <p>III. Contenido</p> <p>I. Introducción</p> <ol style="list-style-type: none"> <li>a. Procesadores de lenguaje</li> <li>b. Estructura de un compilador                             <ol style="list-style-type: none"> <li>i. Análisis léxico</li> <li>ii. Análisis sintáctico</li> <li>iii. Análisis semántico</li> <li>iv. Generación de código intermedio</li> <li>v. Optimización de código</li> <li>vi. Generación de código</li> </ol> </li> <li>c. Evolución de los lenguajes de programación                             <ol style="list-style-type: none"> <li>i. Avance a los lenguajes de alto nivel</li> <li>ii. Impacto de un compilador</li> </ol> </li> <li>d. La ciencia de construir un compilador                             <ol style="list-style-type: none"> <li>i. Modelado en el diseño e implementación de compiladores</li> <li>ii. La ciencia de la optimización de código</li> </ol> </li> <li>e. Aplicaciones de la tecnología de compiladores                             <ol style="list-style-type: none"> <li>i. Implementación de lenguajes de programación</li> <li>ii. Optimización arquitectura computadoras</li> <li>iii. Diseño de nuevas arquitecturas</li> <li>iv. Traducciones de programas</li> <li>v. Herramientas de productividad de software</li> </ol> </li> <li>f. Fundamentos de los lenguajes de programación                             <ol style="list-style-type: none"> <li>i. Distinción entre estático y dinámico</li> <li>ii. Entornos y estados</li> <li>iii. Alcance estático y estructura de bloques</li> <li>iv. Control de acceso explícito</li> <li>v. Alcance dinámico</li> <li>vi. Mecanismo para el paso de parámetros</li> <li>vii. Uso de alias</li> </ol> </li> </ol> <p>II. Análisis léxico</p> <ol style="list-style-type: none"> <li>a. La función del analizador léxico                             <ol style="list-style-type: none"> <li>i. Tokens, patrones y lexemas</li> <li>ii. Atributos de los tokens</li> <li>iii. Errores léxicos</li> </ol> </li> <li>b. Uso de buffer en la entrada                             <ol style="list-style-type: none"> <li>i. Pares de búferes y centinelas</li> </ol> </li> <li>c. Especificación de los tokens                             <ol style="list-style-type: none"> <li>i. Cadenas y lenguajes</li> <li>ii. Operaciones en los lenguajes</li> <li>iii. Expresiones regulares</li> <li>iv. Definiciones regulares</li> <li>v. Extensiones de las expresiones regulares</li> </ol> </li> <li>d. Reconocimiento de tokens                             <ol style="list-style-type: none"> <li>i. Diagrama de transición de estados</li> <li>ii. Reconocimiento de palabras reservadas e identificadores</li> </ol> </li> </ol>		

- iii. Finalización del bosquejo
    - iv. Arquitectura de un analizador léxico
  - e. Autómatas finitos (PRÁCTICO)
    - i. Autómatas finitos no deterministas
    - ii. Tablas de transiciones
    - iii. Aceptación de las cadenas de entrada mediante los autómatas
    - iv. Autómatas finitos deterministas
  - f. De las expresiones regulares a los autómatas (PRÁCTICO)
    - i. Conversión de un AFN a AFD
    - ii. Simulación de un AFN
    - iii. Eficiencia de la simulación de un AFN
    - iv. Construcción de una AFN a partir de una expresión regular
  - g. Diseño de un generador de analizadores léxicos
    - i. La estructura del analizador generado
    - ii. Coincidencia de patrones con base en los AFN
    - iii. AFD para analizadores léxicos
    - iv. Implementación del operador de preanálisis
  - h. Optimización de los buscadores de concordancia
    - i. Estados significativos de una FN
    - ii. Funciones calculadas a partir del árbol sintáctico
    - iii. Cálculo de anulable, primerapos, y ultimapos
    - iv. Cálculo de siguientes
    - v. Conversión directa de una expresión regular a un AFD
    - vi. Minimización del número de estados de una AFD
    - vii. Minimización de estados en los analizadores léxicos
    - viii. Intercambio de tiempo por espacio en la simulación de un AFD
- III. Análisis sintáctico
  - a. Introducción
    - i. La función del analizador sintáctico
    - ii. Representación de gramáticas
    - iii. Manejo de errores sintácticos
    - iv. Estrategias para recuperarse de los errores
  - b. Gramáticas libres de contexto
    - i. Definición formal y notación
    - ii. Árboles de análisis sintáctico y derivaciones
    - iii. Ambigüedad
    - iv. Verificación del lenguaje generado
    - v. Comparación entre gramáticas y expresiones
  - c. Escritura de una gramática
    - i. Comparación entre análisis léxico y sintáctico
    - ii. Eliminación de la ambigüedad
    - iii. Eliminación de la recursividad por la izquierda
    - iv. Factorización por la izquierda
    - v. Construcción de lenguajes no tipo 2
  - d. Análisis sintáctico descendente
    - i. Análisis sintáctico de descenso recursivo
    - ii. Primero y siguiente
    - iii. Gramáticas LL(1)
    - iv. Análisis sintáctico predictivo no recursivo
    - v. Recuperación de errores en el análisis sintáctico predictivo
  - e. Análisis sintáctico ascendente
    - i. Reducciones
    - ii. Poda
    - iii. Análisis sintáctico de desplazamiento-reducción
    - iv. Conflictos
  - f. Introducción al análisis sintáctico LR Simple
    - i. Elementos y el autómata LR(0)
    - ii. Algoritmo de análisis sintáctico LR
    - iii. Construcciones de tablas de análisis sintáctico SLR
    - iv. Prefijos viables
  - g. Analizadores sintácticos LR poderosos
    - i. Elementos LR(1) canónicos
    - ii. Construcción de conjuntos de elementos LR(1)
    - iii. Tablas de análisis sintáctico LR(1) canónico
    - iv. Construcción de tablas de análisis sintáctico LALR
    - v. Construcción eficiente de tablas LALR
    - vi. Compactación de tablas LR
  - h. Uso de gramáticas ambiguas
    - i. Precedencia y asociatividad para resolver conflictos
    - ii. Ambigüedad del else colgante
    - iii. Recuperación de errores en el análisis sintáctico LR

#### IV. Metodología

Clase magistral para explicación de teoría.

Resolución de tareas, problemas y autoestudio

Tareas de investigación

Proyectos de programación

V. Evaluación:

Clase (60 puntos)		
	3 Exámenes parciales (10 puntos c/u)	30
	Examen final	25
	Tareas y cortos	05
	Total Clase	60
Laboratorio (40 puntos)		
	Tareas y cortos	05
	Proyectos	35
	Total Laboratorio	40

VI. Observaciones

El curso y el laboratorio se aprueban con 61 puntos.

Sitio del curso: <http://ecys.ingenieria-usac.edu.gt/UV/index.php>

Las notas y tareas del curso también serán publicadas en la UV.

Para tener derecho a exámenes parciales y final es necesario cumplir con el 80% de la asistencia, así como para solicitar reposición de exámenes parciales (sólo se da reposición de exámenes parciales).

Exámenes Parciales Martes o Jueves (Secciones B y C) y Sábados (Sección A)

**Primer Examen Parcial Jueves 20/Febrero/2020 – Sábado 22/Febrero/2020**

Incluye

- i. Introducción a la compilación
- ii. Análisis Léxico
  - d. Reconocimiento de tokens

**Segundo Examen Parcial Sábado 21/Marzo/2020 – Martes 24/Marzo/2020**

Incluye

- iii. Análisis Sintáctico
  - c. Escritura de una gramática

**Tercer Examen Parcial Sábado 25/Abril/2020 – Martes 28/Abril/2020**

Incluye

**Resto del contenido**

7	Bibliografía	Libro de Texto Organización de lenguajes y compiladores 1 Aho, lam, Sethi y Ullman. (2008). Compiladores. 2 ed. (Capítulos 1, 3 y 4)
9	Catedráticos titulares	Ing. Manuel Castillo Ing. Keving Lajpop Ing. Mario Bautista