Procesadores del Lenguaje

Curso Académico: (2022 / 2023) **Fecha de revisión: 30-05-2022**

Departamento asignado a la asignatura: Departamento de Informática

Coordinador/a: GARCIA HERRERO, JESUS

Tipo: Obligatoria Créditos ECTS : 6.0

Curso : 3 Cuatrimestre : 2

REQUISITOS (ASIGNATURAS O MATERIAS CUYO CONOCIMIENTO SE PRESUPONE)

Programación (Curso: primero - Cuatrimestre: primero)
Tería de Autómatas y Lenguajes Formales (Curso: segundo - Cuatrimestre: primero)

COMPETENCIAS Y RESULTADOS DEL APRENDIZAJE

¿ Conocer las técnicas de análisis léxico, sintáctico y semántico, y saber desarrollar un analizador para un lenguaje de programación o un lenguaje específico de dominio.
¿ Conocer las técnicas más usuales de generación y optimización de código, y saber determinar el impacto de las mismas.
¿ Conocer y saber aplicar las técnicas más comunes para la detección y recuperación de errores.

DESCRIPCIÓN DE CONTENIDOS: PROGRAMA

Descriptores: Representación de lenguajes, Análisis léxico, Análisis sintáctico, Análisis Semántico, Generación de código, Recuperación de errores, Optimización de código

TEMA I: Introducción
Historia de los compiladores y lenguajes
Conceptos básicos
Lenguajes y Gramáticas
Definiciones formales de Gramática, Expresiones Regulares y Autómata
Fases y estructura de un compilado
Diagramas de Tombston

TEMA II: Análisis Léxico
Diseño de un Analizador Léxico
Autómatas Finitos reconocedores de Lenguajes Regulares
Construcción de un Autómata Finito. Ejemplos
Generadores Automáticos de Analizadores Léxicos: LEX
Manejo de Errores Léxicos

TEMA III: Análisis Sintáctico
Introducción al Análisis Sintáctico
Clasificación de los métodos de Análisis Sintáctico
Análisis Descendente, Análisis Sintáctico LL
Obtención de la tabla LL(1). Ejemplos
Análisis Ascendente, Análisis Sintáctico LR
Tratamiento de Gramáticas Ambiguas. Ejemplos
Generadores automáticos de Analizadores Sintácticos: YACC

TEMA IV: Tratamiento de Errores Sintáctico
Errores. Estrategias de Detección y Recuperación. Ejemplos
Recuperación con diferentes analizadores
Analizador descendente L
Analizador ascendente de precedencia de operador
Analizador ascendente L

TEMA V: Análisis Semántico
Gramáticas de Atributos, Ejemplos, Formalización
Especificación de un traductor: Traducción Dirigida por Sintaxis y Esquemas de Traducción
Evaluación de gramáticas

Construcción de árboles de sintaxis abstracta

TEMA VI: Verificación de Tipo
Introducció
Expresiones de tipo
Sistemas de tipos. Comprobación estática y dinámico
Ejemplo de construcción y verificación de tipos sencillo
Equivalencia de expresiones de tipos
Sobrecarga y Orientación a Objetos

TEMA VII: Generación de Código Intermedio
Tipos de Lenguajes Intermedios
Códigos de tres direcciones. Alternativas
Generación de código intermedio: declaraciones, expresiones aritméticas, arrays
Sentencias de flujo de control

TEMA VIII: Generación de Código Máquina
Código máquina y máquina objetivo
Opciones de código máquina
Instrucciones y direccionamiento y coste
Generación simple de código a partir de lenguaje intermedio
Bloques básicos y grafos de flujo
Asignación de registro
Traducción de otras instrucciones

TEMA IX: Tabla de Símbolos y Entorno de Ejecución
Asignación de memoria
Asignación estática y dinámica
Gestión de pila y montículo. Ejemplos
Llamadas a funciones
Registros de activación
Paso de parámetros
Operaciones y organización de la tabla de símbolos

TEMA X: Optimización de Código
Concepto de Optimización de código
Optimización local sobre bloques básicos
Transformaciones que preservan la función
Eliminación de código inactivo
Optimización de bucles
Análisis global del flujo de datos

TEMA XI: Aspectos Específicos
Otros procesadores de lenguajes
Intérpretes
Preprocesadores y macroprocesadores
Diseño de lenguajes
Estructuras de datos y de control
Aspectos de compilación para tipos específicos de lenguajes
Algunos ejemplos de compiladores

ACTIVIDADES FORMATIVAS, METODOLOGÍA A UTILIZAR Y RÉGIMEN DE TUTORÍAS

Clases Teóricas: 1.5 ECTS. Tienen por objetivo alcanzar las competencias específicas cognitivas de la asignatura.
Clases Prácticas: 1.5 ECTS. Desarrollan las competencias específicas instrumentales y la mayor parte de las transversales, como son la de trabajo en equipo, capacidad de aplicar los conocimientos a la práctica, de planificar y organizar y de análisis y síntesis. También tienen por objetivo desarrollar las capacidades específicas actitudinales. Consisten en el diseño y desarrollo de un proyecto de compilador/intérprete elaborado en grupos de trabajo
Realización de Actividades Académicas Dirigidas
- Con presencia del profesor: 1 ECTS Planteamiento de un estudio, orientado por el profesor pero propuesto por el alumno, donde profundiza sobre algún aspecto de la materia, realizando una exposición pública del mismo.
- Sin presencia del profesor: 1.5 ECTS. Ejercicios y lecturas complementarias propuestas por el profesor.

Ejercicios y Examen: 0.5 ECTS. Tienen por objeto incidir y complementar en el desarrollo de las capacidades específicas cognitivas y procedimentales.

SISTEMA DE EVALUACIÓN

Los ejercicios y exámenes además de servir como actividad formativa tienen el doble objetivo de ser medida para el sistema de evaluación. El sistema de evaluación incluye la valoración de las actividades académicas dirigidas y prácticas según la siguiente ponderación. (No se especifica la relación con las competencias dado que las actividades formativas ya han sido relacionadas con ellas.)
Ejercicios y Examen: 40%
Práctica: 40%
Actividades Académicas Dirigidas:
Con presencia del profesor: 15%
Sin presencia del profesor: 5%

Peso porcentual del Examen Final:	40
Peso porcentual del resto de la evaluación:	60

BIBLIOGRAFÍA BÁSICA

- A. V. Aho and Ravi Sethi and J. D. Ullman Compiladores: Principios, Técnicas y Herramientas, Addison-Wesley Iberoamericana, 1990.
- Kenneth C. Louden Construcción de Compiladores. Principios y práctica, Thomson, 2004.

BIBLIOGRAFÍA COMPLEMENTARIA

- A. V. Aho and J. D. Ullman Principles of Compiler Design, Addison-Wesley, Reading, Mass., 1977.
- C. N. Fisher, R. J. Leblanc Crafting a Compiler with C, Addison-Wesley, 1991.
- Dick Grune, Henri E. Bal, Ceriel J.H. Jacobs, Koen G. Langendoen Modern Compiler Design, John Wiley & Sons, 2000.
- Doug Brown, John Levine, Tony Mason Lex & Yacc, O'Reilly Media, Inc., 1995.
- F. J. Sanchis and C. Galán Compiladores: Teoría y Construcción, Paraninfo, 1986.
- Garrido, Iñesta, Moreno, Pérez Diseño de Compiladores, Publicaciones Universidad de Alicante, 2002.
- K. A. Lemone Fundamentals of Compilers: An Introduction to Computer Language Translation, CRC Press, 1992.
- T. Pittman and J. Peters The Art of Compiler Design: Theory and Practice, Prentice-Hall, Englewood Cliffs, New Jersey, 1992.