ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA

GUIA DOCENTE

CURSO 2017/2018

Grado en Ingeniería Informática itinerario Computación

DATOS DE LA ASIGNATURA				
Nombre:				
Procesadores de Lenguajes				
Denominación en inglés:				
Language processors				
Código:		**Carácter:**		
606010235		Obligatorio		
Horas:				
	Totales	**Presenciales**	**No presenciales**	
Trabajo estimado:	150	60	90	
Créditos:				
	Grupos reducidos			
Grupos grandes	**Aula estándar**	**Laboratorio**	**Prácticas de campo**	**Aula de informática**
3	0	0	0	3
Departamentos:		**Áreas de Conocimiento:**		
Tecnologías de la Información		Ciencia de la Computación e Inteligencia Artificial		
Curso:		**Cuatrimestre:**		
3º - Tercero		Segundo cuatrimestre		

DATOS DE LOS PROFESORES			
Nombre:	**E-Mail:**	**Teléfono:**	**Despacho:**
*Aranda Corral, Gonzalo A.	gonzalo.aranda@dti.uhu.es	87663	TU-7

*Profesor coordinador de la asignatura

Consultar los horarios de la asignatura

DATOS ESPECÍFICOS DE LA ASIGNATURA

1. Descripción de contenidos

1.1. Breve descripción (en castellano):

• Introducción a los procesadores de lenguajes.
• Análisis léxico.
• Análisis sintáctico. Análisis descendente. Análisis ascendente.
• Análisis semántico.
• Organización de la memoria en tiempo de ejecución.
• Generación de código.

1.2. Breve descripción (en inglés):

• Introduction to language processors.
• Lexical analysis.
• Parsing. Descendant parsing. Ascendant parsing.
• Semantic analysis.
• Runtime memory organization.
• Code Generation.

2. Situación de la asignatura

2.1. Contexto dentro de la titulación:

Asignatura de introducción al diseño de compiladores y el procesamiento de lenguajes formales.
Es obligatoria dentro de la especialidad de Computación. Permite a los alumnos profundizar desde las bases teóricas de la Computación (autómatas) hasta sus conocimientos más prácticos sobre lenguajes de programación.

2.2. Recomendaciones:

Es recomendable tener superada la asignatura "Algoritmica y Modelos de Computación", ya que se utilizan conceptos de gramáticas y autómatas que se imparten en dicha asignatura.
Es recomendable tener buenos conocimientos de programación ("Estructuras de Datos", "Metodología de la Programación").

3. Objetivos (Expresados como resultados del aprendizaje):

- Comprender los conceptos fundamentales de la traducción y la interpretación.
- Conocer las fases del proceso de compilación.
- Entender la conexión de los lenguajes de programación con la arquitectura de los procesadores.
- Saber utilizar las herramientas para la generación automática de compiladores e intérpretes.

4. Competencias a adquirir por los estudiantes

4.1. Competencias específicas:

- **CE2-C:** Capacidad para conocer los fundamentos teóricos de los lenguajes de programación y las técnicas de procesamiento léxico, sintáctico y semántico asociadas, y saber aplicarlas para la creación, diseño y procesamiento de lenguajes.

4.2. Competencias básicas, generales o transversales:

- **CB5:** Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía
- **CG0:** Capacidad de análisis y síntesis: Encontrar, analizar, criticar (razonamiento crítico), relacionar, estructurar y sintetizar información proveniente de diversas fuentes, así como integrar ideas y conocimientos.
- **G02:** Capacidad de comunicación oral y escrita en el ámbito académico y profesional con especial énfasis, en la redacción de documentación técnica
- **G03:** Capacidad para la resolución de problemas
- **G04:** Capacidad para tomar decisiones basadas en criterios objetivos (datos experimentales, científicos o de simulación disponibles) así como capacidad de argumentar y justificar lógicamente dichas decisiones, sabiendo aceptar otros puntos de vista
- **G05:** Capacidad de trabajo en equipo.
- **G06:** Capacidad para el aprendizaje autónomo así como iniciativa y espíritu emprendedor
- **T01:** Uso y dominio de una segunda lengua
- **T02:** Conocimiento y perfeccionamiento en el ámbito de las TIC’s

5. Actividades Formativas y Metodologías Docentes

5.1. Actividades formativas:

- Sesiones de Teoría sobre los contenidos del Programa.
- Sesiones de Resolución de Problemas.
- Sesiones Prácticas en Laboratorios Especializados o en Aulas de Informática.

5.2. Metologías docentes:

- Clase Magistral Participativa.
- Desarrollo de Prácticas en Laboratorios Especializados o Aulas de Informática en grupos reducidos.
- Resolución de Problemas y Ejercicios Prácticos.
- Planteamiento, Realización, Tutorización y Presentación de Trabajos.
- Evaluaciones y Exámenes.

5.3. Desarrollo y justificación:

- Sesiones académicas de teoría

Las clases teóricas tendrán una duración de 2 horas. En ellas se expondrá y explicará, con ayuda del cañón de proyecciones y/o la pizarra los contenidos asociados a cada tema. Habrá bibliografía específica de cada tema disponibles en la web de la asignatura con antelación suficiente.

- Sesiones académicas de problemas

Al finalizar las sesiones de teoría de cada tema se desarrollarán las sesiones de problemas correspondientes al tema desarrollado. Para cada tema de teoría se facilitará un boletín de problemas. En estas sesiones se resolverán los problemas más representativos de cada boletín.

- Sesiones de prácticas de laboratorio

Las sesiones de prácticas se desarrollarán en aulas provistas de ordenadores y tendrán una duración de 2 horas. En estas prácticas se explicarán aspectos de implementación de las diferentes fases de un compilador. El código a explicar en cada una de las sesiones estará disponible en la web de la asignatura con la suficiente antelación. En estas sesiones se utilizará Java como lenguaje de programación, Eclipse como entorno de desarrollo, JavaCC como herramienta de generación de analizadores y alguna herramienta de simulación de microprocesadores (de código libre)

- Resolución y entrega de problemas/prácticas

A lo largo del curso se planteará uno trabajo práctico a desarrollar por los alumnos de manera individual. El trabajo se referirá al desarrollo de un procesador de lenguaje sin la ayuda de herramientas automáticas, tal y como se explica en el primer bloque de prácticas. Este trabajo se considera una actividad académica dirigida y su explicación se realizará en el horario de las sesiones de prácticas. El seguimiento de este trabajos se realizará en tutorías individualizadas.

6. Temario desarrollado:

- Temario de teoría

Tema 1: Introducción.
1.1 Conceptos
1.2 Un poco de historia
1.3 Estructura de un compilador
1.4 Teoría de lenguajes formales
Tema 2: Análisis léxico.
2.1 Introducción
2.2 Especificación de categorías léxicas
2.3 Autómatas Finitos No Deterministas
2.4 Autómatas Finitos Deterministas
2.5 Implementación de un analizador léxico
Tema 3: Análisis sintáctico descendente.
3.1 Características del análisis sintáctico
3.2 Gramáticas libres de contexto
3.3 Análisis descendente con retroceso
3.4 Análisis descendente predictivo
3.5 La condición LL(1)
3.6 Analizadores descendentes dirigidos por tabla
3.7 Analizadores descendente recursivos
3.9 Tratamiento de errores
Tema 4: Análisis sintáctico ascendente.
4.1 Análisis sintáctico por desplazamiento y reducción
4.2 El autómata reconocedor de prefijos viables
4.3 Algoritmos LR(0) y SLR
4.4 Tratamiento de errores
4.5 Clasificación de gramáticas
Tema 5: Análisis semántico.
5.1 Características del análisis semántico
5.2 Gramáticas atribuidas
5.3 Especificación de un traductor
5.4 Traductores descendentes
5.5 Traductores ascendentes
5.6 Estructuras básicas
Tema 6: Organización y gestión de la memoria.
6.1 Organización de la memoria en tiempo de ejecución
6.2 Zona de código
6.3 Memoria estática
6.4 Memoria de pila
Tema 7: Generación de código.
7.1 Visión general
7.2 Código de tres direcciones
7.3 Código asociado a las instrucciones comunes

- Temario de prácticas

Práctica 1: Características generales del lenguaje Tinto.
Práctica 2: Implementación del analizador léxico del lenguaje Tinto
Práctica 3: La herramienta JavaCC. Especificación léxica.
Práctica 4: Analizadores sintácticos descendentes.
Práctica 5: La herramienta JavaCC. Especificación sintáctica.
Práctica 6: Tratamiento de errores.
Práctica 7: Analizadores sintácticos ascendentes.
Práctica 8: El árbol de sintaxis abstracta de Tinto.
Práctica 9: Análisis semántico en el compilador de Tinto.
Práctica 10: La herramienta JavaCC. Especificación semántica.
Práctica 11: Organización de la memoria en el compilador de tinto.
Práctica 12: Generación de código intermedio en el compilador de Tinto.
Práctica 13: Generación de código objeto en el compilador de Tinto.

7. Bibliografía

7.1. Bibliografía básica:

A.V. Aho, R. Sethi, J.D. Ullman, “Compiladores: Principios, Técnicas y Herramientas”. Addison-Wesley Iberoamérica, 1990. ISBN: 0-201-62903-8.
A.W. Appel, “Modern Compiler Implementation in Java (2nd edition)”. Cambridge University Press, 2002. ISBN: 0-521-82060-X.
A. Garrido Alenda, y otros . “Diseño de compiladores”. Servicio de publicaciones de la Universidad de Alicante, 2002. ISBN: 84-7908-700-5.

7.2. Bibliografía complementaria:

K.C. Louden, “Construcción de compiladores. Principios y práctica”. Thomson editores, 2004. ISBN: 970-686-299-4.
S. Muchnick, “Advanced Compiler Design and Implementation”. Morgan Kaufmann Publishers, San Francisco California, 1997. ISBN: 1-55860-320-41
B. Teufel, S. Schmidt, T. Teufel, “Compiladores. Conceptos fundamentales”. Addison-Wesley Iberoamericana, 1995. ISBN: 0-201-65365-6.
S. Gálvez Rojas, M.A. Mora Mata, "Compiladores: traductores y compiladores con lex/yacc, jflex/cup y javacc", Universidad de Málaga, 2005. (edición electrónica gratuita)
D. Grune, H.E. Bal, C.J.H. Jacobs, K.G. Langendoen, "Modern Compiler Design", Ed. Wiley, 2000.
R. Wilhelm, D. Maurer, "Compiler Design", Ed. Addison-Wesley, 1995.

8. Sistemas y criterios de evaluación.

8.1. Sistemas de evaluación:

- Examen de teoría/problemas
- Examen de prácticas

8.2. Criterios de evaluación y calificación:

Examen de teoría/problemas: 50%
Examen de prácticas: 50%
Es necesaria una calificación mínima de 4.0 puntos (sobre 10.0) en cada una de las partes para calcular la calificación. En cualquier otro caso, la calificación será de 0 o no presentado, según corresponda.

9. Organización docente semanal orientativa:

Semanas	Grupos Grandes	Grupos Reducidos Aula Estándar	Grupos Reducidos Aula de Informática	Grupos Reducidos Laboratorio	Grupos Reducidos prácticas de campo	Pruebas y/o actividades evaluables	Contenido desarrollado
#1	2	0	2	0	0		Presentación
#2	2	0	2	0	0		Tema 1
#3	2	0	2	0	0		Tema 2
#4	2	0	2	0	0		Tema 2
#5	2	0	2	0	0	Publicación trabajo final a entregar	Tema 3
#6	2	0	2	0	0		Tema 3
#7	2	0	2	0	0		Tema 4
#8	2	0	2	0	0		Tema 4
#9	2	0	2	0	0		Tema 4
#10	2	0	2	0	0		Tema 5
#11	2	0	2	0	0		Tema 5
#12	2	0	2	0	0		Tema 5
#13	2	0	2	0	0		Tema 6
#14	2	0	2	0	0	Entrega del trabajo	Tema 7
#15	2	0	2	0	0	Fuera de Calendario	
	30	0	30	0	0		