

Grado en Ingeniería Informática

DATOS DE LA ASIGNATURA

Nombre:

Diseño de Compiladores

Denominación en inglés:

Compilers Design

Código:

606010309

Carácter:

Optativo

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:

Tecnologías de la Información

Áreas de Conocimiento:

Ciencia de la Computación e Inteligencia Artificial

Curso:

4º - Cuarto

Cuatrimestre:

Primer cuatrimestre

DATOS DE LOS PROFESORES

Nombre:

*Moreno Velo, Francisco
José

E-Mail:

francisco.moreno@dti.uhu.es

Teléfono:

87659

Despacho:

Edificio Torreumbria,
despacho 14.

*Profesor coordinador de la asignatura

1. Descripción de contenidos

1.1. Breve descripción (en castellano):

Introducción a los Compiladores
 Tabla de símbolos.
 Árbol de Sintáxis Abstracta
 Generación de código intermedio
 Organización y gestión de la memoria
 Optimización de código
 Técnicas básicas de optimización
 Optimización local
 Generación de código final
 Paralelización de programas secuenciales
 Aspectos específicos de los LOO
 Aspectos específicos de los LF

1.2. Breve descripción (en inglés):

Introduction to Compilers
 Table of Symbols.
 Abstract Syntax Tree.
 Intermediate code generation.
 Memory management and organization.
 Code Optimization.
 Basic techniques.
 Local optimization.
 Object code generation.
 Parallelization of sequential programs.
 Specific aspects of OOL.
 Specific aspects of FL.

2. Situación de la asignatura

2.1. Contexto dentro de la titulación:

Asignatura optativa de 4 curso de la titulación donde se profundiza en las técnicas para el diseño adecuado de compiladores y traductores.

2.2. Recomendaciones:

Se recomienda para esta asignatura tener conocimientos de programación orientada a objetos y estructuras de datos.

3. Objetivos (Expresados como resultados del aprendizaje):

Conocer e implementar las diferentes fases del proceso de compilación de los lenguajes imperativos: análisis, gestión de memoria y generación de código.
 Conocer e implementar las técnicas básicas de optimización de código.
 Conocer las técnicas básicas de gestión de memoria dinámica.

4. Competencias a adquirir por los estudiantes

4.1. Competencias específicas:

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
- **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 Prácticas en Laboratorios Especializados o en Aulas de Informática.
- Actividades Académicamente Dirigidas por el Profesorado: seminarios, conferencias, desarrollo de trabajos, debates, tutorías colectivas, actividades de evaluación y autoevaluación.

5.2. Metodologías docentes:

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

5.3. Desarrollo y justificación:

Se desarrollarán clases teóricas donde se impartirán los conceptos fundamentales de la asignatura y sus aplicaciones. Estos conceptos se reforzarán en unas series de prácticas que deberán realizar en grupos reducidos. Finalmente se realizará una prueba donde el alumno deberá de reflejar los conocimientos adquiridos tanto a nivel teórico como práctico.

6. Temario desarrollado:

TEMARIO TEÓRICO

Tema 1: Introducción a los Compiladores

- 1.1 Definiciones
- 1.2 Etapas de compilación

Tema 2: Análisis léxico

- 2.1 Características
- 2.2 Análisis léxico basado en Autómatas No Deterministas
- 2.3 Contextos léxicos

Tema 3: Análisis sintáctico

- 3.1 Características
- 3.2 Análisis descendente con gramáticas extendidas
- 3.3 Lookahead adaptativo
- 3.4 Análisis ascendente LR(1)
- 3.5 Análisis ascendente LALR

Tema 4: Tabla de símbolos.

- 4.1 Operaciones frecuentes
- 4.2 Integración semántica
- 4.3 Gestión de ámbitos
- 4.4 Reglas de visibilidad: Importación, Exportación
- 4.5 Declaraciones implícitas
- 4.6 Sobrecarga
- 4.7 Referencias adelantadas

Tema 5: Análisis semántico

- 5.1 Características
- 5.2 Estructura del Árbol de Sintaxis Abstracta
- 5.3 Construcción (recursiva) del ASA

Tema 6: Generación de código intermedio

- 6.1 Lenguajes Intermedios: Tipos
- 6.2 Código dos direcciones
- 6.3 Código tercetos indirectos
- 6.4 Código tres direcciones
- 6.5 Código asociado a las instrucciones comunes
- 6.6 Código para máquinas abstractas

Tema 7: Organización y gestión de la memoria

- 7.1 Organización de la memoria en tiempo de ejecución
- 7.2 Zona de código, memoria estática y memoria de pila
- 7.3 Memoria con reserva dinámica
- 7.4 Recolección de basura

Tema 8: Optimización de código

- 8.1 Optimización .vs. óptimo
- 8.2 Etapas de optimización
- 8.3 Dependencia de la máquina.

Tema 9: Técnicas básicas de optimización

- 9.1 Cálculo previo de constantes
- 9.2 Reducción de fuerza y frecuencia
- 9.3 Optimización de ciclos
- 9.4 Eliminación de código
- 9.5 Reacomodos

Tema 10: Optimización local

- 10.1 Bloques Básicos
- 10.2 Grafo Dirigido Acíclico
- 10.3 Eliminación código redundante
- 10.4 Expresiones comunes
- 10.5 Alojamiento en registros

Tema 11: Generación de código final

- 11.1 Resolución de referencias simbólicas
- 11.2 Modos de direccionamiento
- 11.3 Entornos de ejecución
- 11.4 Selección de instrucciones

Tema 12: Aspectos avanzados

- 12.1 Paralelización de programas secuenciales
- 12.2 Aspectos específicos de los lenguajes orientados a objetos
- 12.3 Aspectos específicos de los lenguajes funcionales

TEMARIO PRÁCTICO

Práctica 1: Completar el conjunto de instrucciones del lenguaje Tinto

Práctica 2: Añadir operadores de bit

Práctica 3: Añadir operadores de incremento, decremento y asignación

Práctica 4: Añadir tipos de datos enteros (byte, short y long)

Práctica 5: Añadir tipos de datos en coma flotante (float y double)

Práctica 6: Optimización por paso de argumentos por registro

Práctica 7: Optimización por reordenación de código

Práctica 8: Optimización por alojamiento en registro
Práctica 9: Añadir tipo de dato array
Práctica 10: Liberación de memoria con contadores de uso
Práctica 11: Constantes y variables globales
Práctica 12: Añadir clases

7. Bibliografía

7.1. Bibliografía básica:

Libro: .Compiladores. Principios, Técnicas y Herramientas.
Autores: A.V. Aho.
Editorial: Addison-Wesley Iberoamericana. Año: 1998
Libro: .Compiladores e Intérpretes: teoría y práctica.
Autores: Manuel Alfonseca Morena y otros
Editorial: Pearson . Prentice Hall. Año: 2006
Libro: .Modern compiler implementation in Java(second edition).
Autores: Andrew W. Appel
Editorial: Cambridge. Año: 2002

7.2. Bibliografía complementaria:

Libro: .Construcción de compiladores.
Autores: Kenneth C. Louden
Editorial: Thomson Año: 2004
Libro: .Compiladores. Teoría y Construcción.
Autores: Sanchís Llorca y Galán Pascual
Editorial: Paraninfo Año: 1986
Libro: .Compiladores e Intérpretes. Un enfoque pragmático.
Autores: Sanchez Dueñas y Valverde Andreu
Editorial: Diaz de Santos, S.A. Año: 1984
Libro: .Compiladores. Conceptos fundamentales.
Autores: Teufel, Schmidt y Teufel
Editorial: Addison-Wesley Iberoamericana Año: 1995

8. Sistemas y criterios de evaluación.

8.1. Sistemas de evaluación:

- Examen de teoría/problemas
- Defensa de Trabajos e Informes Escritos
- Seguimiento Individual del Estudiante
- Examen de prácticas

8.2. Criterios de evaluación y calificación:

La evaluación de la asignatura consta de una parte teórica y una parte práctica.
La parte teórica se evalúa por medio de las convocatorias oficiales de exámenes (junio y septiembre).
La parte práctica se evalúa por medio de un trabajo individual.
Las calificaciones serán ponderadas según la siguiente relación: 50% examen, 40% trabajo final y 10% participación en clase.
El alumno deberá superar en, al menos, un 40% cada una de las partes y obtener un mínimo del 50% de la media aritmética de las partes.

9. Organización docente semanal orientativa:

	<i>Semanas</i>	<i>Grupos Grandes</i>	<i>Grupos Reducidos Aula Estándar</i>	<i>Grupos Reducidos Aula de Informática</i>	<i>Grupos Reducidos Laboratorio</i>	<i>Grupos Reducidos prácticas de campo</i>	Pruebas y/o actividades evaluables	Contenido desarrollado
#1	2	0	2	0	0		Presentación, Tema 1	
#2	2	0	2	0	0		Tema 2	
#3	2	0	2	0	0		Tema 3	
#4	2	0	2	0	0		Tema 4	
#5	2	0	2	0	0		Tema 4	
#6	2	0	2	0	0		Tema 5	
#7	2	0	2	0	0		Tema 6	
#8	2	0	2	0	0		Tema 7	
#9	2	0	2	0	0		Tema 7	
#10	2	0	2	0	0		Tema 8	
#11	2	0	2	0	0		Tema 9	
#12	2	0	2	0	0		Tema 9	
#13	2	0	2	0	0		Tema 10	
#14	2	0	2	0	0		Tema 11	
#15	2	0	2	0	0		Tema 12	
	30	0	30	0	0			