



Guía Docente				
Datos Identificativos			2012/13	
Asignatura (*)	Compiladores	Código	614451111	
Titulación				
Descriptorios				
Ciclo	Período	Curso	Tipo	Créditos
Mestrado Oficial	Anual	Primeiro	Obrigatoria	5
Idioma	CastelánGalego			
Prerrequisitos				
Departamento	Tecnoloxías da Información e as Comunicaci3ns			
Coordinaci3n	Dafonte Vazquez, Jose Carlos	Correo electr3nico	carlos.dafonte@udc.es	
Profesorado	Dafonte Vazquez, Jose Carlos	Correo electr3nico	carlos.dafonte@udc.es	
Web	lia2.tic.udc.es/compiladores			
Descrici3n xeral	<p>Compiladores;cc tradutores e intérpretes; etapas de un compilador; optimizaci3n de c3digo; macroprocesadores. O obxectivo 3 familiarizar 3 alumno co funcionamento dos compiladores, o entorno no que traballan as3 coma algunhas ferramentas software para a construcci3n dos mesmos.Asumar a caracter3stica interdisciplinar da asignatura. Adquirir os coñecementos necesarios para deseñar e implementar as diferentes etapas necesarias para o desenvolvemento dun compilador: an3lise (l3xico, sint3ctico e sem3ntico) e s3ntese (xeraci3n de c3digo intermedio, optimizaci3n de c3digo e xeraci3n de c3digo obxeto).</p>			

Competencias da titulaci3n	
C3digo	Competencias da titulaci3n

Resultados da aprendizaxe			
Competencias de materia (Resultados de aprendizaxe)	Competencias da titulaci3n		
Coñecer e comprender as t3cnicas utilizadas para o deseño de compiladores. Deseñar e implementar as diferentes fases necesarias para o deseño dun compilador. Coñecer e manexar algunhas das ferramentas m3is habituais neste campo.	AP2	BP1	CM1
	AP5	BP2	CM3
	AP7	BP3	CM6
	AP8	BP4	CM7
	AP9	BP5	CM8
	AP11	BP6	
	AP12	BP7	
		BP8	
		BP10	
		BP11	
		BP12	
		BP15	

Contidos	
Temas	Subtemas
Tema I. Introducci3n	1.1 Estructura de un compilador. 1.2 Ejemplo de las fases de un compilador.



Tema II. Lenguajes y Gramáticas	<ul style="list-style-type: none">2.1 Notación de Chomsky.2.2 Clasificación de Chomsky.2.3 Gramáticas de contexto libre (GCL).2.4 Diagramas de Conway.2.5 Reglas BNF.2.6 Problemas en las GCL.2.7 Simplificación de gramáticas.2.8 Gramática limpia.2.9 Forma normal de Chomsky (FNC).2.10 Resumen.2.11 Ejercicios.
Tema III. Análisis Léxico	<ul style="list-style-type: none">3.1 Tipos de máquinas reconocedoras o autómatas.3.2 Autómatas Finitos.3.3 Conversión de una Gramática Regular en un Autómata finito.3.4 Expresión regular.3.5 Algoritmo de Thompson.3.6 Transformación de un AFND-lambda en un AFD.3.7 Traductores finitos (TF).3.8 Implementación de autómatas.<ul style="list-style-type: none">3.8.1 Tabla compacta.3.8.2 Autómata programado.3.9 Ejemplo. Scanner para números reales sin signo en Pascal.3.10 Acciones semánticas.3.11 Generador LEX.
Tema IV. Análisis sintáctico (Parsing)	<ul style="list-style-type: none">4.1 Máquinas teóricas, mecanismos con retroceso<ul style="list-style-type: none">4.1.1 Autómatas con pila (AP).4.1.2 Esquemas de traducción (EDT).4.1.3 Traductores con pila (TP).4.2 Algoritmos sin retroceso.<ul style="list-style-type: none">4.2.1 Análisis sintáctico ascendente por precedencia simple.4.2.2 Análisis sintáctico ascendente por precedencia de operadores.4.2.3 Analizadores descendentes LL(K).4.2.4 Analizadores ascendentes LR(k).4.2.5 Generador de analizadores sintácticos YACC.
Tema V. Análisis semántico	<ul style="list-style-type: none">5.1 Definiciones dirigidas por la sintaxis.5.2 Esquema de traducción.5.3 Comprobaciones en tiempo de compilación.
Tema VI. Generación de código	<ul style="list-style-type: none">6.1 Lenguajes intermedios.6.2 Generación de código intermedio.6.3 Generación de código desde lenguaje intermedio.
Tema VII. Optimización de código	<ul style="list-style-type: none">7.1 Algoritmo de Nakata.7.2 Un ejemplo de optimización manual.7.3 Lazos en los grafos de flujo.7.4 Análisis global del flujo de datos.7.5 Solución iterativa de las ecuaciones de flujo de datos.
Tema VIII. Errores	<ul style="list-style-type: none">8.1 Tipos de errores.8.2 Recuperación de errores léxico-gráficos.
Tema IX. Intérpretes y complementos	<ul style="list-style-type: none">9.1 Estructura de un intérprete actual.9.2 Arquitectura neutral de java.



Planificación

Metodoloxías / probas	Horas presenciais	Horas non presenciais / traballo autónomo	Horas totais
Sesión maxistral	40	40	80
Proba obxectiva	3	3	6
Prácticas de laboratorio	15	15	30
Atención personalizada	9	0	9

*Os datos que aparecen na táboa de planificación son de carácter orientativo, considerando a heteroxeneidade do alumnado

Metodoloxías

Metodoloxías	Descrición
Sesión maxistral	Sesións maxistrais coincidentes entre a asignatura do Master e en Enxeñería Informática. O axuste de horas non é exacto pero, de cara a facilitar o desenvolvemento do proxecto, no segundo cuatrimestre dispoñeráse dunha redución de horas maxistrais a tal efecto, igualando o esquema de docencia entre ambas.
Proba obxectiva	Faranse dous exames, un en febreiro/marzo sobre o contido do primeiro cuatrimestre e outro en xuño do contido do segundo cuatrimestre. Neste último tamén haberá unha segunda oportunidade para recuperar, se é necesario, a materia do primeiro cuatrimestre.
Prácticas de laboratorio	Realizaranse varias pequenas prácticas hasta xaneiro e a partir de ahí haberá que desenvolver un proxecto que involucre a maior parte das fases dun compilador. Este proxecto será proposto polo estudante.

Atención personalizada

Metodoloxías	Descrición
Prácticas de laboratorio	Especialmente no caso do proxecto a desenvolver polo alumno, realizarase un seguimento semanal dos traballos.

Avaliación

Metodoloxías	Descrición	Cualificación
Proba obxectiva	Duas probas (febreiro/marzo e xuño) que aportarán cada unha un 50% da nota neste apartado (será condición imprescindible ter presentadas as pequenas prácticas iniciais).	60
Prácticas de laboratorio	Proxecto a propoñer e desenvolver polo alumno durante o segundo cuatrimestre	40
Outros		

Observacións avaliación

En calquera caso, é preciso aprobar as dúas parte. No caso contrario, a máxima nota que se poderá acadar é un 4.5.
--

Fontes de información

Bibliografía básica	
Bibliografía complementaria	

Recomendacións

Materias que se recomenda ter cursado previamente

Materias que se recomenda cursar simultaneamente

Deseño de Sistemas Operativos/614407114

Enxeñería do Software/614407115

Materias que continúan o temario

Linguaxes Naturais/614407220

Observacións



A asignatura troncal de Enxeñería Informática e Enxeñería Técnica en Informática de Sistemas "Teoría de autómatas e linguaxes formais" é de gran utilidade para a comprensión da asignatura de Compiladores.

(*A Guía docente é o documento onde se visualiza a proposta académica da UDC. Este documento é público e non se pode modificar, salvo casos excepcionais baixo a revisión do órgano competente dacordo coa normativa vixente que establece o proceso de elaboración de guías