Proyecto/Guía docente de Lenguajes de Programación

Asignatura	LENGUAJES DE PROGRAMACIÓN		
Materia	COMPUTACIÓN		
Módulo	COMPLEMENTOS DE FORMACIÓN		
Titulación	MÁSTER EN INGENIERÍA INFORMÁTICA		
Plan	545	**Código**	53162
Periodo de impartición	2º CUATRIMESTRE	**Tipo/Carácter**	OB
Nivel/Ciclo		**Curso**	
Créditos ECTS	6 ECTS		
Lengua en que se imparte	CASTELLANO		
Profesor/es responsable/s	M. LUISA GONZÁLEZ DÍAZ		
Datos de contacto (E-mail, teléfono…)	mluisa@infor.uva.es 983 423000 ext. 5615		
Departamento	INFORMÁTICA (ATC, CCIA, LSI)		

1. Situación / Sentido de la Asignatura

1.1 Contextualización

Los lenguajes de programación desempeñan un papel central en la formación de cualquier graduado en Ingeniería Informática. Después de haber adquirido una cierta competencia en la programación y en la solución algorítmica de problemas y conocidos distintos paradigmas y lenguajes, el paso natural es analizar por qué los lenguajes de programación se diseñan tal y como los conocemos.

El estudio de los lenguajes de programación conecta los aspectos teóricos sobre los que se fundamenta la computación con el origen de la solución algorítmica de problemas y con las arquitecturas sobre las que se ejecutan los programas, así como con los aspectos metodológicos relacionados con la producción de software y servicios de calidad.

La estructura y propiedades de las diversas familias de lenguajes de programación guardan además una relación directa con la forma en que se representa y procesa la información usando computadoras y redes de computadoras, lo que constituye el núcleo conceptual y estructural de la disciplina.

Esta asignatura se organiza en torno a la propuesta de unidades de conocimiento propuestas en el Computing Curricula 2001 de la ACM en relación con los Lenguajes de Programación y se orienta más al análisis de la estructura y funcionamiento de los lenguajes de programación que a la revisión del inmenso catálogo de familias y ejemplares de lenguajes existentes. Con ello, se sientan las bases para comprender con mayor profundidad las características y posibilidades que ofrecen los diversos lenguajes de programación que el graduado tenga que afrontar a lo largo de su vida profesional y, al mismo tiempo, se establece un punto de partida para el diseño de lenguajes, lo que sin duda puede tener una importancia capital para el desarrollo de software y servicios en el futuro.

1.2 Relación con otras materias

En consonancia con todo ello, esta asignatura se ofrece en el tercer curso del grado en Ingeniería Informática, orientado a un perfil profesional de desarrollo de software y servicios de carácter general. Aparece después de las asignaturas de Fundamentos de Programación (1º), Paradigmas de Programación (1º), Interacción Persona Computadora (2º), Fundamentos de Ingeniería del Software (2º) y Programación Orientada a Objetos (2º), materias con las que está directamente relacionada. Sigue también a otras materias relacionadas con la plataforma tecnológica, como Fundamentos de Computadoras (1º), Fundamentos de Sistemas Operativos (2º), Fundamentos de Redes (1º) o Sistemas Distribuidos (2º), cuya superación favorecerá sin duda una mayor facilidad en la consecución de las competencias básicas asociadas a esta asignatura.

1.3 Prerrequisitos

Aunque no se establezcan como condiciones previas, los alumnos que cursen esta asignatura deberían haber adquirido las competencias básicas de programación asociadas a las materias de Fundamentos de Programación, Paradigmas de Programación, Programación Orientada a Objetos y las competencias básicas metodológicas que proporcionan las asignaturas de Fundamentos de Ingeniería del Software y de Interacción Persona Computadora. Se supone también un conocimiento de los aspectos arquitectónicos básicos contenidos en las asignaturas de Fundamentos de Computadores y de Arquitectura y Organización de Computadores.

2. Competencias

2.1 Generales

Código	Descripción
CG2	Capacidad para dirigir las actividades objeto de los proyectos del ámbito de la informática de acuerdo con los conocimientos adquiridos según lo establecido en las competencias de formación especificadas a continuación en esta sección de la memoria.
CG3	Capacidad para diseñar, desarrollar, evaluar y asegurar la accesibilidad, ergonomía, usabilidad y seguridad de los sistemas, servicios y aplicaciones informáticas, así como de la información que gestionan
CG4	Capacidad para definir, evaluar y seleccionar plataformas hardware y software para el desarrollo y la ejecución de sistemas, servicios y aplicaciones informáticas, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias de formación específicas indicadas en la sección 3.1 de la memoria de Grado
CG5	Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería de software como instrumento para el aseguramiento de su calidad, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias de formación específicas indicadas en la sección 3.1 de la memoria de Grado
CG6	Capacidad para concebir y desarrollar sistemas o arquitecturas informáticas centralizadas o distribuidas integrando hardware, software y redes de acuerdo con los conocimientos adquiridos según lo establecido en las competencias de específicas indicadas en la sección 3.1 de la memoria de Grado
CG8	Conocimiento de las materias básicas y tecnologías, que capaciten para el aprendizaje y desarrollo de nuevos métodos y tecnologías, así como las que les doten de una gran versatilidad para adaptarse a nuevas situaciones.
CG9	Capacidad para resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad. Capacidad para saber comunicar y transmitir los conocimientos, habilidades y destrezas de la profesión de Ingeniero Técnico en Informática
CG10	Conocimientos para la realización de mediciones, cálculos, valoraciones, tasaciones, peritaciones, estudios, informes, planificación de tareas y otros trabajos análogos de informática, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias de específicas indicadas en la sección 3.1 de la memoria de Grado

Transversales:

Código	Descripción
CT1	Capacidad de análisis y síntesis
CT2	Capacidad de organizar y planificar
CT3	Comunicación oral y escrita en la lengua propia
CT4	Capacidad para la lectura de textos técnicos en inglés
CT5	Habilidades de gestión de la información
CT6	Resolución de problemas
CT7	Toma de decisiones
CT8	Capacidad crítica y autocrítica
CT9	Trabajo en equipo
CT10	Capacidad de trabajar en un equipo interdisciplinar
CT11	Responsabilidad y compromiso ético
CT12	Liderazgo
CT13	Capacidad de aplicar los conocimientos en la práctica
CT14	Capacidad de aprender
CT15	Capacidad de adaptarse a nuevas situaciones
CT16	Habilidad para trabajar de forma autónoma
CT17	Iniciativa y espíritu emprendedor

2.2 Específicas

Comunes a la rama de Informática

Código	Descripción
CI8	Capacidad para analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados.

De la mención Ingeniería de Software

Código	Descripción
IS4	Capacidad para identificar y analizar problemas y diseñar, construir, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales.

3. Objetivos

Código	Descripción
	Conocer los fundamentos de la teoría de lenguajes formales y su conexión con la programación.
	Entender los principios de diseño de los lenguajes de programación y saber cómo relacionar sus componentes estructurales y funcionales básicos.
	Ser capaz de construir analizadores léxicos y sintácticos para lenguajes de programación sencillos.
	Ser capaz de diseñar analizadores semánticos para lenguajes de programación sencillos.
	Saber aplicar el conjunto de restricciones de diseño impuestas por el entorno de programación y ejecución concreto de un lenguaje de programación.
	Conocer los principios y técnicas básicas de la optimización local y global de código.

4. Contenidos y/o bloques temáticos

Bloque 1: "Fundamentos de Procesamiento de Lenguajes"

Carga de trabajo en créditos ECTS: 4'0

a. Contextualización y justificación

En este bloque se sientan las bases formales y prácticas de la construcción de procesadores de lenguajes. Se introducirán los aspectos básicos del proceso de compilación e interpretación desde un punto de vista funcional, así como los rudimentos de la teoría de lenguajes formales que sirvan para fundamentar y resolver cada fase de la división del procesamiento en niveles léxico, sintáctico, semántico y generación de código.

En el apartado práctico, se desarrollarán actividades para familiarizar al estudiante con las expresiones regulares, el concepto de compilación y sus fases, y la codificación de máquinas reconocedoras. A continuación se presentarán, en sucesivas sesiones, entornos de generación de analizadores estándar como base para desarrollos posteriores. Los alumnos realizarán en equipo prácticas que desarrollen un ejemplo de cada una de las fases esenciales del proceso.

b. Objetivos de aprendizaje

- Conocer los fundamentos de la teoría de lenguajes formales y su conexión con la programación.
- Ser capaz de construir analizadores léxicos y sintácticos para lenguajes de programación sencillos.
- Ser capaz de diseñar analizadores semánticos para lenguajes de programación sencillos

c. Contenidos

TEORÍA

TEMA 1: Introducción al procesamiento de lenguajes

TEMA 2: Léxico y Sintaxis de los lenguajes de programación

TEMA 3: Análisis Semántico y Generación de Código

TEMA 4: Optimización

SEMINARIOS

- Expresiones regulares
- Compilación
- Codificación de máquinas reconocedoras

LABORATORIOS

- Lex
- Yacc
- ANTLR4

PRÁCTICAS

- P1: Nivel léxico
- P2: Nivel sintáctico
- P3: Nivel semántico

Bloque 2: "Bases de diseño de lenguajes"

Carga de trabajo en créditos ECTS: 2.0

a. Contextualización y justificación

En este segundo bloque enfrentamos las cuestiones cruciales en diseño de lenguajes. Comenzaremos analizando el concepto de "nombre" como abstracción en este contexto y las nociones relacionadas con ello. A continuación nos centramos en la estructura de datos y control y en los mecanismos de abstracción más importantes que se soportan en muchos lenguajes de programación. Se analizan las diferentes variantes de control de flujo presentes en los lenguajes de programación y los tipos de datos, tanto desde un punto de vista descriptivo como desde un punto de vista algebraico, analizando los sistemas de tipos y los problemas de equivalencia y conversión.

En el apartado práctico, se implantarán, en el traductor que se construye con las técnicas del bloque anterior, los conceptos estudiados en éste, incorporando tipos y estructuras de control.

b. Objetivos de aprendizaje

- Entender los principios de diseño de los lenguajes de programación y saber cómo relacionar sus componentes estructurales y funcionales básicos
- Saber aplicar el conjunto de restricciones de diseño impuestas por el entorno de programación y ejecución concreto de un lenguaje de programación

c. Contenidos

TEORÍA

TEMA 1: Aspectos estructurales globales

TEMA 2: Control de flujo

TEMA 3: Tipos de Datos

PRÁCTICAS y LABORATORIO

Integrados en los del bloque anterior

d. Métodos docentes

Ver apartados siguientes

e. Plan de trabajo

f. Evaluación

Ver apartados siguientes

g. Bibliografía básica (ambos bloques)

- Michael L. Scott, *Programming Language Pragmatics*, 3rd. ed., Morgan Kauffmann, 2009. ISBN 978-0-12-374514-9.
- Kenneth E. Louden, Kenneth A. Lambert, *Programming Languages: Principles and Practice*. Course Technology, 2012 ISBN: 9781111529413
- Alfred V. Aho, Monica S. Lam, Ravi Sethi y Jeffrey D. Ullman, *Compilers. Principles, Techniques and Tools*, 2nd ed. Pearson, 2007. ISBN 0-321-48681-1
- Terrence Parr, *The Definitive ANTLR4 Reference*. The Pragmatic Bookshelf, 2012, ISBN-13:978-1-93435-699-9

h. Bibliografía complementaria (ambos bloques)

- Terrence W. Pratt y Marvin V. Zelkowitz, *Lenguajes de Programación: Diseño e Implementación*, 3ª. ed., Prentice-Hall, 1998. ISBN 970-17-0046-5.
- Robert Harper, *Practical foundations for Programming Languages*. Creative Commons License. 2012
- Alfred Aho, Ravi Sethi y Jeffrey Ullman, *Compiladores: Principios, Técnicas y Herramientas*, 2nd ed. Pearson, 2008. ISBN 978-970-26-1133-2.

i. Recursos necesarios

Todos los recursos necesarios para el desarrollo de las actividades teóricas y prácticas estarán a disposición de los alumnos en la página de la asignatura en el campus virtual o referidos en ella.

j. Temporalización

CARGA ECTS	PERIODO PREVISTO DE DESARROLLO

5. Métodos docentes y principios metodológicos

La materia es amplia, y en este curso solamente se sientan sus bases fundamentales, combinando visiones teórica, práctica e instrumental.
Por ello, se emplearán clases magistrales participativas, estudio de casos en aula, resolución de prácticas en laboratorio, aprendizaje de herramientas en laboratorio, desarrollo de varias prácticas de curso por grupos de alumnos y tutorías grupales e individuales.

6. Tabla de dedicación del estudiante a la asignatura

ACTIVIDADES PRESENCIALES	HORAS	ACTIVIDADES NO PRESENCIALES	HORAS
Clases teórico-prácticas	30-3 = **27**		
Laboratorios	**24**		
Seminarios (S)	**6**		
Evaluación (fuera del periodo oficial de exámenes)	3·1 = **3**		
		Estudio y trabajo autónomo individual	**65**
		Estudio y trabajo autónomo grupal	**25**
Total presencial	**60**	**Total no presencial**	**90**

7. Sistema y características de la evaluación

INSTRUMENTO/PROCEDIMIENTO	PESO EN LA NOTA FINAL	OBSERVACIONES
Seminarios Cuestionarios (1 hora)	15%	Primeras sesiones en horario de laboratorios
Entregas de prácticas	25%	Grupales. A lo largo del curso y entrega final
Cuestionarios (1 hora)	10%	Materia cubierta hasta el momento. En sesión de aula, anunciada en su momento
Examen escrito	50%	Período de exámenes

CRITERIOS DE CALIFICACIÓN

Convocatoria ordinaria:
Para superar la asignatura deberán cumplirse todas las condiciones siguientes:

- Superar una calificación mínima en la parte de entregas de ejercicios prácticos
- Superar una calificación mínima en el examen escrito
- Obtener una calificación final resultante igual o mayor que 5 con las ponderaciones citadas.

Convocatoria extraordinaria:

- **Se conservan las calificaciones de las entregas de prácticas y de los seminarios, con sus ponderaciones**
- **Se realizará un examen escrito con un peso del 65%**
- **Para superar la asignatura será preciso cumplir todas las condiciones siguientes:**
  - Superar una calificación mínima en el examen escrito de segunda convocatoria.

○ Si las entregas de prácticas en convocatoria ordinaria no superan el mínimo, se propondrá una nueva entrega práctica. Su enunciado se comunicará finalizada la convocatoria ordinaria, tendrá relación con las realizadas durante el curso y su calificación podrá tener en cuenta la conseguida en éstas. Se obtendrá una calificación cuyo peso será de nuevo el 25% ○ Obtener una calificación final resultante igual o mayor que 5 con las ponderaciones especificadas

8. Consideraciones finales

El alumno deberá estar atento durante el curso a la página de la asignatura, donde podrá encontrar el material necesario y los detalles que complementarán esta guía.