GUÍA DOCENTE DE PROYECTO DE AUTÓMATAS Y LENGUAJES

La presente guía docente corresponde a la asignatura Proyecto de Autómatas y Lenguajes (PAyL) del Grado en Ingeniería Informática, aprobada para el curso lectivo 2017-2018 en Junta de Centro y publicada en su versión definitiva en la página web de la Escuela Politécnica Superior. La guía docente de PAyL aprobada y publicada antes del periodo de matrícula tiene el carácter de contrato con el estudiante.

ASIGNATURA

PROYECTO DE AUTÓMATAS Y LENGUAJES (PAyL)

1.1. Código

17839 de la titulación de Grado en Ingeniería Informática

1.2. Materia

Autómatas y Lenguajes

1.3. Tipo

Formación obligatoria

1.4. Nivel

Grado

1.5. Curso

3º Ingeniería Informática, 4º plan conjunto Informática/Matemáticas

1.6. Semestre

1º

1.7. Número de créditos

3 créditos ECTS

1.8. Requisitos previos

Para un buen aprovechamiento del curso, es recomendable haber aprobado las asignaturas *Programación I*, *Programación II* y *Proyecto de programación* de la materia *Programación* del módulo *Programación y Estructura de datos*. De no ser así, el curso puede seguirse pero requiriendo de un considerable esfuerzo extra lo que puede incidir en el rendimiento del estudiante en otras asignaturas en las que esté matriculado.

El contenido de la asignatura también tiene una relación aunque más marginal con la materia *Análisis de Algoritmos*, del módulo *Programación y estructuras de datos*, que se imparte como asignatura con el mismo nombre; *Seminario Taller de software*, del mismo módulo y que forma parte de la materia *Seminarios Taller de Informática*; y finalmente con la materia *Estructuras Discretas y Lógica*, del módulo *Fundamentos teóricos de la informática y aplicaciones*.

La asignatura *Proyecto de Autómatas y Lenguajes* se imparte en el primer semestre del tercer curso en el Grado de Informática. Conforma, junto con la asignatura *Autómatas y Lenguajes* la materia de *Autómatas y Lenguajes* (*materia 2*) del módulo *Fundamentos teóricos de la informática y aplicaciones*. Ambas asignaturas se imparten en el mismo semestre. Por tanto, es imprescindible el buen aprovechamiento en cada una de ellas para superar con éxito ambas. Esta asignatura también se imparte en el primer semestre del cuarto curso de la titulación conjunta en Informática/Matemáticas

Se ha realizado un esfuerzo considerable en la producción de material docente propio de las asignaturas para que resulte lo más completa y útil posible al alumno. Se podrá acceder a este material a través de la plataforma Moodle (http://uam-virtual.es)
Este material incluirá para el Proyecto de Autómatas y Lenguajes

- Una copia de las trasparencias en las que se apoyarán las explicaciones de los profesores.
- Algunas librerías auxiliares para facilitar la generación de código ensamblador (nasm)

Se requiere, por parte del alumno, iniciativa personal y constancia en el trabajo cotidiano en todas las fases del proyecto.

1.9. Requisitos mínimos de asistencia a las sesiones presenciales

Esta asignatura dispone de dos métodos de evaluación: continua y no continua. Los estudiantes deberán cumplir los distintos modelos de evaluación que conlleva cada uno de los métodos publicados en la presente guía docente (ver apartado 4).

En la evaluación continua la asistencia es obligatoria al menos en un 80%. En la evaluación no continua la asistencia es muy recomendable aunque no obligatoria.

1.10. Datos del equipo docente

Dra. Marina de la Cruz Echeandía (coordinadora)
Departamento de Ingeniería Informática
Escuela Politécnica Superior
Despacho - Módulo: B-407-3 Edificio B – 4ª Planta
Teléfono: +34 497 3364
Correo electrónico: marina.cruz@uam.es
Página web: http://arantxa.ii.uam.es/~mdlcruz
Horario de atención al alumnado: Petición de cita previa por correo electrónico.

1.11. Objetivos del curso

Las **competencias** específicas de Computación que se pretenden adquirir con esta asignatura son las siguientes:

CC2. 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.

Los objetivos que se pretende alcanzar con esta asignatura son:

OBJETIVOS GENERALES	
G1	Conocer representaciones intermedias, como los ensambladores (en concreto NASM).
G2	Saber diseñar la gramática de atributos compatible con analizadores sintácticos ascendentes para la construcción de compiladores de una pasada para lenguajes de programación de alto nivel con diferentes características capaces de traducir el lenguaje fuente a código ensamblador nasm
G3	Saber desarrollar en el lenguaje de programación C las tablas de símbolos para el lenguaje objeto del proyecto (con estructura de bloques) y el tipo de compilador objeto del proyecto (de una pasada)
G4	Saber utilizar las herramientas de diseño flex y bison y el lenguaje de programación C para el desarrollo del compilador del lenguaje
G5	Saber programar casos de prueba con el lenguaje de programación objeto del proyecto que permitan la verificación de la corrección del compilador

OBJETIVOS ESPECÍFICOS POR TEMA	
TEMA 0.- Introducción a los procesadores de lenguaje	
1	Conocer la función de los procesadores de lenguaje, sus tipos y estructura
TEMA 1.- Representaciones intermedias	
2	Conocer el ensamblador NASM y saber escribir código en dicho lenguaje.
TEMA 2.- Diseño de un lenguaje de programación	
3	Conocer las construcciones de las gramáticas independientes del contexto con las que describir las construcciones sintácticas del lenguaje de programación objeto del proyecto
4	Determinar con claridad los aspectos del lenguaje de programación que no se pueden expresar mediante reglas independientes del contexto
TEMA 3.- Diseño de la tabla de símbolos	
5	Saber definir la estructura de la tabla de símbolos adecuada para el lenguaje y el compilador que se está desarrollando (lenguaje con estructura de bloques y compilador de una pasada) basándose en tablas hash
TEMA 4.- Diseño del analizador morfológico del lenguaje de programación	
6	Saber determinar el fragmento de la gramática del lenguaje del que se ocupará el analizador morfológico
7	Saber diseñar los patrones regulares necesarios para la implementación del analizador morfológico con la herramienta que se va a utilizar (flex)
TEMA 5.- Diseño del analizador sintáctico del lenguaje de programación	
8	Saber expresar la gramática en el formato de la herramienta que se va a utilizar (bison)
9	Saber solucionar los conflictos de análisis que puedan aparecer
10	Saber definir la asociatividad y prioridad en los operadores especialmente para gramáticas ambiguas en cuanto a las expresiones aritméticas
11	Saber unificar los analizadores morfológicos y sintácticos desarrollados
TEMA 6.- Analizador semántico y generador de código	
12	Saber traducir las restricciones semánticas del lenguaje de programación en un sistema de atributos para el tipo de compilador que se está diseñando (compilador de una pasada basado en un analizador sintáctico ascendente)
13	Saber modificar la gramática independiente del contexto del lenguaje de programación para incorporar el sistema de atributos
14	Conocer diferentes alternativas para la gestión del espacio auxiliar para gestionar las expresiones aritméticas en código ensamblador
15	Conocer diferentes protocolos de comunicación de parámetros y valores de retorno de las subrutinas en código ensamblador
16	Conocer las técnicas de gestión de bajo nivel para las estructuras de datos, las estructuras de control de flujo del programa, la entrada/salida y la gestión de memoria

17	Saber modificar la gramática del lenguaje de programación para incorporar la generación de código de las diferentes construcciones del lenguaje de programación

1.12. Contenidos del programa

PROGRAMA SINTÉTICO

El objetivo de esta asignatura es la realización de un proyecto informático completo en el ámbito de los procesadores de lenguaje, en concreto, la construcción de un compilador para un lenguaje de programación sencillo pero también completo. Para ello el alumno utilizará herramientas informáticas de ayuda adecuadas al nivel de conocimientos que posee (flex y bison con lenguaje de programación C y ensamblador NASM).
Para ello debe completarse los contenidos de la asignatura *Autómatas y Lenguajes* con temas dedicados a la generación de código (directa y representaciones intermedias) así como nociones de optimización.

Tema 0: Introducción a los procesadores de lenguaje

Tema 1: Lenguaje ensamblador NASM

Tema 2: Especificación del lenguaje de programación ALFA

Tema 3: Tablas de símbolos

Tema 4: Diseño e implementación del analizador morfológico

Tema 5: Diseño e implementación del analizador sintáctico (ascendente)

Tema 6: Diseño e implementación de analizador semántico y generador de código

PROGRAMA DETALLADO

Tema 0: Introducción a los procesadores de lenguaje

Tema 1: Lenguaje ensamblador NASM

Tema 2: Especificación del lenguaje de programación ALFA

- Especificación de las características del lenguaje para el cual se va a desarrollar un compilador.

Tema 3: Tablas de símbolos

- Elección del tipo de tabla de símbolos (habitualmente compiladores de una pasada con estructura de bloques).
- Diseño e implementación (lenguaje de programación C) de la tabla de símbolos para el lenguaje de programación.

Tema 4: Diseño e implementación del analizador morfológico
- Delimitación de la parte de la gramática independiente del contexto que será responsabilidad del analizador morfológico.
- Diseño de patrones regulares para el analizador morfológico.
- Diseño e implementación del analizador morfológico utilizando herramientas informáticas de ayuda (flex y lenguaje de programación C).

Tema 5: Diseño e implementación del analizador sintáctico (ascendente)
- Diseño e implementación del analizador sintáctico utilizando herramientas informáticas de ayuda (bison y lenguaje de programación C).
- El punto anterior implica:
- Reescribir la gramática del lenguaje en el formato de entrada de las herramientas (bison).
- Análisis y solución de conflictos en el analizador.
- Garantizar la correcta comunicación con el analizador morfológico.

Tema 6: Diseño e implementación del analizador semántico y generador de código
- Uso de atributos semánticos en herramientas de ayuda al desarrollo de compiladores completos (bison y lenguaje de programación C)
- Análisis semántico y generación directa de código para el lenguaje e programación

1.13. Referencias de consulta

Bibliografía:

Nota: Las referencias de consulta de esta asignatura son compartidas con la asignatura de Autómatas y Lenguajes dada la conexión que hay entre ellas y que se ha explicado previamente. La documentación disponible en Moodle contiene suficiente detalle como para suponer una referencia de consulta completa. Por lo tanto, la consulta de estos materiales, aunque puede resultar recomendable para un dominio más profundo de la materia, no es imprescindible para alcanzar los objetivos y competencias descritos en esta guía.

1. Alfonseca, M., de la Cruz, M., Ortega, A., Pulido, E. Compiladores e intérpretes: teoría y práctica, Pearson, 2006.
2. Aho, A.V., Sethi, R., Ullman, J.D., Compilers: Principles, Techniques and Tools. Addison-Wesley Publishing Company, Reading, MA, 1986. Traducción español, Compiladores: principios, técnicas y herramientas, Pearson Education, 1990-98.
3. Grune, D., Modern Compiler Design, Wiley, 2000.

4. Marcotty, M., Ledgard, H.F., Bochmann, G.V., A Sampler of Formal Definitions. Computing Surveys, 8:2, pp. 191-276, June 1976.
5. Alfonseca, M., Alfonseca, E., Moriyon, R. Teoría de Autámatas y Lenguajes Formales, McGraw Hill, 2007. ISBN: 978-84-481-5637-4.
6. Alfonseca, M., Sancho, J., Martínez Orga, M.A., Teoría de Lenguajes, Gramáticas y Autómatas. Universidad y Cultura, Madrid, 1987.

Nota: no se recomienda a los estudiantes comprar ningún libro hasta no haber comparado su contenido con el programa y revisado previamente en la biblioteca.

2. Métodos docentes

La metodología utilizada en el desarrollo de la actividad docente incluye los siguientes tipos de actividades:

- **Contenidos de carácter teórico:**

Fundamentalmente el tema 0.

Actividad del profesor

Clases expositivas simultaneadas con la realización de problemas y ejercicios teórico-prácticos. Se utilizarán los recursos audiovisuales (proyector de trasparencias) e informáticos disponibles en el laboratorio.

Actividad del estudiante:

Actividad presencial: Toma de apuntes, participación activa en clase respondiendo a las cuestiones planteadas.

Actividad no presencial: estudio del material proporcionado y resolución de los problemas planteados en clase.

- **Contenidos de carácter práctico:**

El proyecto, consistente en el desarrollo completo de un compilador utilizando el siguiente soporte informático:

- Sistema operativo Linux.
- Lenguaje de programación C.
- Herramienta de ayuda al desarrollo de analizadores morfológicos flex.
- Herramienta de ayuda al desarrollo de procesadores de lenguaje bison (+flex).
- Software público para la gestión de ensamblador nasm.
- Herramientas de desarrollo, depuración y compilación en C. Los alumnos podrán utilizar las que consideren oportunas.

El proyecto se desarrollará de manera individual. No obstante, si el número de alumnos por grupo es superior al número de equipos disponibles en los laboratorios, será necesario el desarrollo del proyecto en grupos.

Cada alumno/grupo debe realizar a lo largo del curso una serie de entregas para poder asegurar la conclusión del compilador a final de curso y además, poder evaluar la evolución de su trabajo (consúltese la fecha prevista para las entregas en el cronograma). En concreto las entregas serán las siguientes:

1. Tabla de símbolos
2. Analizador morfológico
3. Analizador sintáctico
4. Compilador completo

Cada una de las entregas será evaluada por el profesor y permitirá establecer un proceso de realimentación con el alumno. Las notas obtenidas serán utilizadas para el cálculo de la nota final como se especifica en la sección 4.

Al final de curso, los alumnos entregarán el compilador que será evaluado y al que se le asignará una calificación numérica entre 0 y 10 que también será utilizada para el cálculo de la nota final como se especifica en la sección 4.

Por último, los alumnos deben presentarse a un examen final que se realizará de manera individual en el laboratorio y que consistirá, en general, en la realización de modificaciones al compilador. Los alumnos obtendrán una nota entre 0 y 10 en este examen que será utilizada para el cálculo de la nota final como se indica en la sección 4.

A lo largo del proyecto se planifican 3 posibles tutorías, en función de las necesidades, para facilitar el seguimiento del trabajo.

Actividad del profesor

Clases expositivas de los contenidos necesarios para la realización de cada una de las partes del compilador. Sugerencia de posibles alternativas a los problemas que aparezcan en el diseño del compilador. Realimentación a cada alumno/grupo de manera individual sobre su trabajo en cada entrega. Se utilizarán los recursos audiovisuales (proyector de trasparencias) e informáticos disponibles en el laboratorio.

Actividad del estudiante:

Actividad presencial: Participación activa en clase respondiendo a las cuestiones planteadas. Asistir a las tutorías a las que le convoque su profesor.

Actividad no presencial: estudio del material proporcionado, planificación individual o en grupo del desarrollo del trabajo para garantizar la superación de todas las entregas y la superación de la materia. Este trabajo podría incluir el diseño del conjunto de tareas que requiere cada entrega, y si procede, su reparto entre los miembros del grupo, y el trabajo en grupo para la integración

de cada parte con las reuniones necesarias para que todos los miembros del grupo dominen el trabajo realizado.
Es también responsabilidad de los alumnos tomar la iniciativa en la solicitud de tutorías que pudieran necesitar ante las dificultades que encuentren en cualquier momento del proyecto.

***Tutorías presenciales:**
Se han planificado 3 tutorías de 1 hora para los alumnos de evaluación continua coincidiendo con los momentos más exigentes para los alumnos. En ellas se intentará focalizar la atención de los alumnos en los aspectos más relevantes de la materia y minimizar el número de dudas que queden sin resolver.

Actividad del profesor:
Organización de las tutorías presenciales.

Actividad del alumno:
Actividad presencial: Asistir a la tutoría, plantear sus dudas y atender a las explicaciones de su profesor.
Actividad no presencial: Previo a la tutoría, identificación de posibles dudas con relación al temario de la tutoría.

3. Tiempo de trabajo del estudiante

Se asignan 25 horas de trabajo a cada crédito europeo, por tanto una asignatura de 3 créditos conlleva 75 horas de trabajo del estudiante, que incluyen tanto tareas presenciales como no presenciales. En la Tabla 1 se indica el porcentaje de cada actividad respecto al total de 75 horas para los estudiantes que han seleccionado el método de evaluación continua con asistencia obligatoria a clase en relación con las directrices del Espacio Europeo de Educación Superior.

Tabla 1. Distribución del Tiempo de Trabajo del Estudiante en la Asignatura Proyecto de Autómatas y Lenguajes

		Nº de horas	Porcentaje
Presencial	Clases teóricas	14 h (18,67%)	49.3% = 37 horas
	Clases prácticas	14 h (18,67%)	
	Tutorías programadas a lo largo del semestre	3 h (4%)	
	Realización exámenes finales	6= 3h+3h(finales) (8%)	
No presencial	Realización de actividades prácticas	25 h (33,3%)	50.7% = 38 horas
	Preparación examen final (conv. ord.)	5 h (6.7%)	
	Preparación examen final (conv. extraord.)	8 h (10.7%)	

Carga total de horas de trabajo: 25 horas x 3 ECTS	**75 h**

4. Métodos de evaluación y porcentaje en la calificación final

Convocatoria ordinaria

EVALUACIÓN CONTINUA

Es necesario cumplir con el requisito de asistencia a clase, al menos el 80% de las sesiones. Los alumnos que no cumplan ese requisito pasarán a la evaluación no continua desde el momento que lo incumplan.

Los alumnos realizarán diferentes actividades correspondientes a las diferentes partes del compilador: la tabla de símbolos, el analizador morfológico y el analizador sintáctico. Se evaluarán todas las actividades propuestas por los profesores (entregas, cuestionarios, etc). A las calificaciones de estas tres partes, entre 0 y 10, las llamaremos TS, AM y AS. Se realizará una media ponderada de estas tres notas para obtener la calificación que llamaremos INTERMEDIA. Se utilizará la siguiente fórmula:

$$INTERMEDIA = 0,3*TS + 0,3*AM + 0,4*AS$$

Los alumnos cuya nota intermedia no sea igual o superior a 5 pasarán a la evaluación no continua.

A final de curso, los alumnos entregarán el compilador y obtendrán una calificación entre 0 y 10 que a partir de ahora llamaremos COMPILADOR. Esta nota deberá ser igual o superior a 5 para poder obtener posteriormente la nota final de la convocatoria ordinaria, es decir, si un alumno suspende la entrega del compilador, no podrá aprobar en la convocatoria ordinaria.

Los alumnos cuya nota INTERMEDIA y COMPILADOR sea igual o superior a 5, a final de curso realizarán un examen y obtendrán una calificación entre 0 y 10. A esta calificación la llamaremos EXAMEN. Esta nota deberá ser igual o superior a 5 para poder obtener la nota final de la convocatoria ordinaria.

La nota final de cada alumno en la convocatoria ordinaria se calculará de acuerdo a la siguiente fórmula:

$$0,4*\text{INTERMEDIA} + 0,4*\text{COMPILADOR} + 0,2*\text{EXAMEN}$$

En el caso en que no se pueda obtener una nota final debido a que no se haya cumplido alguno de los requisitos, la nota final en actas será:

$$0,4*\text{Mín}(5,\text{INTERMEDIA}) + 0,4*\text{Mín}(5,\text{COMPILADOR}) + 0,2*\text{Mín}(5,\text{EXAMEN})$$

EVALUACIÓN NO CONTINUA

Los alumnos de evaluación no continua, el mismo día que los alumnos de evaluación continua entreguen el compilador, también lo harán, siendo la especificación del compilador la misma para los dos tipos de evaluación. En la entrega del compilador obtendrán una calificación entre 0 y 10 que a partir de ahora llamaremos COMPILADOR. Esta nota deberá ser igual o superior a 5 para poder obtener posteriormente la nota final de la convocatoria ordinaria, es decir, si un alumno suspende la entrega del compilador, no podrá aprobar en la convocatoria ordinaria.

El mismo día que los alumnos de evaluación continua realicen el examen, los alumnos de evaluación no continua realizarán también un examen que podrá ser diferente del de evaluación continua y obtendrán una calificación entre 0 y 10. A esta calificación la llamaremos EXAMEN. Esta nota deberá ser igual o superior a 5 para poder obtener la nota final de la convocatoria ordinaria.

La nota final de cada alumno en la convocatoria ordinaria se calculará de acuerdo a la siguiente fórmula:

$$0,3*\text{COMPILADOR} + 0,7*\text{EXAMEN}$$

En el caso en que no se pueda obtener una nota final debido a que no se haya cumplido alguno de los requisitos, la nota final en actas será:

$$0,3*\text{Mín}(5,\text{COMPILADOR}) + 0,7*\text{Mín}(5,\text{EXAMEN})$$

Sobre la convocatoria extraordinaria

Para la convocatoria extraordinaria los alumnos deberán entregar un compilador que amplíe las funcionalidades del compilador de la convocatoria ordinaria. En la entrega del compilador obtendrán una calificación entre 0 y 10 que a partir de ahora llamaremos COMPILADOR. Esta nota deberá ser igual o superior a 5 para poder obtener posteriormente la nota final de la convocatoria extraordinaria.

Posteriormente, realizarán el examen del compilador y obtendrán una calificación entre 0 y 10. A esta calificación la llamaremos EXAMEN. Esta nota deberá ser igual o superior a 5 para poder obtener la nota final de la convocatoria extraordinaria.

La nota final de cada alumno en la convocatoria extraordinaria se calculará de acuerdo a la siguiente fórmula:

$$0{,}3*\text{COMPILADOR} + 0{,}7*\text{EXAMEN}$$

En el caso en que no se pueda obtener una nota final debido a que no se haya cumplido alguno de los requisitos, la nota final en actas será:

$$0{,}3*\text{Mín}(5,\text{COMPILADOR}) + 0{,}7*\text{Mín}(5,\text{EXAMEN})$$

Sobre calificación "no evaluado" en evaluación continua

En evaluación continua, el número mínimo de pruebas a las que el estudiante se ha de presentar para recibir una calificación numérica es dos tercios (3) del número máximo de pruebas (5). Por debajo de este mínimo, el estudiante recibirá la calificación "No evaluado".

NOTA IMPORTANTE: El hecho de aprobar la entrega del compilador no garantiza de ninguna manera el aprobado en el examen cuyo contenido puede estar relacionado con cualquiera de las funcionalidades del compilador propuesto para el curso.

5. Cronograma

El cronograma del estudiante que sigue el método de evaluación con asistencia obligatoria a clase se muestra en la Tabla 4. Este cronograma tiene carácter orientativo y le ofrece al estudiante una visión global del curso y le permite planificar su trabajo de forma realista.

Semana	Contenido		Horas no presenciales

		Horas presenciales	
1	- Presentación de la asignatura, normas de evaluación y material en Moodle - Tema 0: Introducción a los procesadores de lenguaje	2	2
2	- Tema 0: Introducción a los procesadores de lenguaje - Tema 1: Lenguaje ensamblador NASM	2	2
3	Tema 1: Lenguaje ensamblador NASM	2	2
4	Tema 1: Lenguaje ensamblador NASM	2	
5	- Tema 3: tabla de símbolos ● Explicación de la tabla de símbolos ● Trabajo de los alumnos en la tabla de símbolos del compilador	2	1
6	- Tema 3: tabla de símbolos ● Trabajo de los alumnos en la tabla de símbolos del compilador	2	2
7	**Entrega de la tabla de símbolos antes del comienzo de la sesión** Tema 4: análisis morfológico ● Explicación del analizador morfológico ● Trabajo de los alumnos en el analizador morfológico	2	2
8	**Entrega del analizador morfológico antes del comienzo de la sesión** Tema 5: análisis sintáctico ● Explicación del analizador sintáctico ● Trabajo de los alumnos en el analizador sintáctico	2	2
9	**Entrega del analizador sintáctico antes del comienzo de la sesión** Tema 6: análisis semántico y generación de código ● Explicación del análisis semántico y la generación de código (parte 1) ● Trabajo de los alumnos en el análisis semántico y la generación de código (parte 1)	2	2
10	Tema 6: análisis semántico y generación de código ● Explicación del análisis semántico y la generación de código (parte 2) ● Trabajo de los alumnos en el análisis semántico y la generación de código (parte 2)	2	2
11	Tema 6: análisis semántico y generación de código	2	2

	● Explicación del análisis semántico y la generación de código (parte 3) ● Trabajo de los alumnos en el análisis semántico y la generación de código (parte 3)		
12	Tema 6: análisis semántico y generación de código ● Explicación del análisis semántico y la generación de código (parte 4) ● Trabajo de los alumnos en el análisis semántico y la generación de código (parte 4)	2	2
13	Tema 6: análisis semántico y generación de código ● Explicación del análisis semántico y la generación de código (parte 5) ● Trabajo de los alumnos en el análisis semántico y la generación de código (parte 5)	2	2
14	Tema 6: análisis semántico y generación de código ● Explicación del análisis semántico y la generación de código (parte 6) ● Trabajo de los alumnos en el análisis semántico y la generación de código (parte 6) **Entrega del compilador completo**	2	2
entre 1-14	- Tutorías (3 de 1 h) con los alumnos	3	
15-16	Examen final (ordinaria)	3	5
17-19	Examen final (extraordinaria)	3	8