Datos Descriptivos

ASIGNATURA:	Profundización en Ingeniería del Software
MATERIA:	Diseño de Sistemas de Información
CRÉDITOS EUROPEOS:	3
CARÁCTER:	Obligatoria
TITULACIÓN:	Máster Universitario en Ingeniería Informática
CURSO/SEMESTRE	Primero
ESPECIALIDAD:	

CURSO ACADÉMICO	**2014-2015**		
PERIODO IMPARTICION	**Septiembre- Enero**	**Febrero - Junio**	
	X		
IDIOMA IMPARTICIÓN	**Sólo castellano**	**Sólo inglés**	**Ambos**
			X

DEPARTAMENTO:	Lenguajes y Sistemas Informáticos e Ingeniería del Software

PROFESORADO		
NOMBRE Y APELLIDO (C = Coordinador)	**DESPACHO**	**Correo electrónico**
Natalia Juristo Juzgado (C)	5104	natalia@fi.upm.es
Sira Vegas Hernández	5105	svegas@fi.upm.es
Oscar Dieste	6203	odieste@fi.upm.es

CONOCIMIENTOS PREVIOS REQUERIDOS PARA PODER SEGUIR CON NORMALIDAD LA ASIGNATURA	
ASIGNATURAS SUPERADAS	
OTROS RESULTADOS DE APRENDIZAJE NECESARIOS	Programación en Java

Objetivos de Aprendizaje

COMPETENCIAS Y NIVEL ASIGNADAS A LA ASIGNATURA		
Código	**COMPETENCIA**	**NIVEL**
CB6	Poseer y comprender conocimientos que aporten una base u oportunidad de ser originales en el desarrollo y/o aplicación de ideas, a menudo en un contexto de investigación	Bajo
CG6	Gestión económica y administrativa	Bajo
CG10	Aplicación de los métodos de resolución de problemas más recientes o innovadores y que puedan implicar el uso de otras disciplinas	Medio
CG14	Apreciación de los límites del conocimiento actual y de la aplicación práctica de la tecnología más reciente	Medio
CE1	Capacidad para la integración de tecnologías, aplicaciones, servicios y sistemas propios de la Ingeniería Informática, con carácter generalista, y en contextos más amplios y multidisciplinares	Medio
CE4	Capacidad para modelar, diseñar, definir la arquitectura, implantar, gestionar, operar, administrar y mantener aplicaciones, redes, sistemas, servicios y contenidos informáticos	Alto
CE8	Capacidad para analizar las necesidades de información que se plantean en un entorno y llevar a cabo en todas sus etapas el proceso de construcción de un sistema de información	Alto
CE19	Capacidad para desarrollar e implantar una solución informática en un entorno empresarial	Alto

Código	**RESULTADOS DE APRENDIZAJE DE LA ASIGNATURA**
RA1. -	Ser capaz de descomponer los requisitos de un sistema en fragmentos (*slices*) implementables de forma directa
RA2. -	Ser capaz de desarrollar código utilizando las estrategias *test-last* y *test-first*
RA3. -	Ser capaz de validar el código utilizando tests automatizados

Contenidos y Actividades de Aprendizaje

CONTENIDOS ESPECÍFICOS (TEMARIO)		
TEMA / CAPITULO	**APARTADO**	**Indicadores Relacionados**
Tema 1: Introducción	1.1 Introducción a *Unit Testing*	I1
	1.2 Frameworks de *Unit Testing*	I1
Tema 2: *Incremental test-last*	2.1 Slicing	I2
	2.2 Principios de *Unit Testing*	I2
Tema 3: *Test-first*	3.1 *Test-Driven Development*	I3
Tema 4: Mocking	4.1 *Test doubles*	I4
	4.2 *Frameworks* de *mocking*	I4
Tema 5: Aspectos avanzados de los frameworks de testing	5.1. Hamcrest *matchers* 5.2. Extensión de funcionalidades	I5

BREVE DESCRIPCIÓN DE LAS MODALIDADES ORGANIZATIVAS UTILIZADAS Y METODOS DE ENSEÑANZA EMPLEADOS	
CLASES DE TEORIA	Las clases constarán de una parte teórica, en las que el profesor presenta los conceptos principales de la asignatura
CLASES PROBLEMAS	El alumno deberá ejercitar dichos conceptos principales mediante problemas resueltos en clase. Esto servirá como refuerzo a las clases teóricas y ayudará al alumno a reforzar los conceptos aprendidos en ella
PRACTICAS	El alumno deberá realizar una serie de prácticas (descritas en el apartado de criterios de calificación) a lo largo del curso
TRABAJOS AUTONOMOS	El alumno debe complementar las clases de teoría y problemas con estudio individual. Asimismo, algunas de las prácticas a realizar deberán ser realizadas de forma individual
TRABAJOS EN GRUPO	La práctica relacionada con el análisis de defectos encontrados (última de ellas) llevará asociada una componente de trabajo en grupo. El alumno deberá discutir con sus compañeros de clase sus resultados
TUTORÍAS	En todo momento, el profesor monitorizará el progreso del alumno. El alumno recibirá siempre que lo necesite cualquier tipo de ayuda académica relacionada con los temas que se tratan en el curso

RECURSOS DIDÁCTICOS	
BIBLIOGRAFÍA	1. Beck, K. Test Driven development by Example. Addison Wesley, 2002.
	2. Osherove, R. The art of Unit testing with examples in C#. Manning, 2nd ed., 2014.
	3. Koskela, L. Efective unit testing: A guide for Java developers. Manning, 1st ed, 2013.
RECURSOS WEB	Sitio Moodle de la asignatura
	Tutoriales de varios *frameworks* de *mocking*
EQUIPAMIENTO	Aula
	Sala Informática

Cronograma de trabajo de la asignatura

Semana	Actividades Aula	Laboratorio	Trabajo Individual	Trabajo en Grupo	Actividades Evaluación	Otros
Semana 1 (2 horas)	Introducción (1 horas)		• Estudio individual (1 horas)			
Semana 2 (10 horas)	Junit (2 horas)		• Estudio individual (4 horas) • Resolución de ejercicios y problemas (4 horas)			
Semana 3 (6 horas)	Slicing (2 horas)		• Estudio individual (1 hora) • Resolución de ejercicios y problemas (2 horas)			
Semana 4 (2 horas)					• Práctica de desarrollo de código *test-last* (2 horas)	
Semana 5 (6 horas)	Principios de *Unit Testing* (2 horas)		• Estudio individual (2 horas) • Resolución de ejercicios y problemas (2 horas)			
Semana 6 (2 horas)					• Práctica de desarrollo de código *incremental test-last* (2 horas)	
Semana 7 (7 horas)	*Test Driven Development* (2 horas)		• Estudio individual (1 hora) • Resolución de ejercicios y problemas (4 horas)			
Semana 8 (2 horas)					• Práctica de desarrollo de código *test-first* (2 horas)	

Semana	Actividades Aula	Laboratorio	Trabajo Individual	Trabajo en Grupo	Actividades Evaluación	Otros
Semana 9 (5 horas)	*Test doubles* (2 horas)		• Estudio individual (1 hora) • Resolución de ejercicios y problemas (2 horas)			
Semana 10 (10 horas)	*Frameworks* de *mocking* (2 horas)		• Estudio individual (4 horas) • Resolución de ejercicios y problemas (4 horas)			
Semana 11 (2 horas)					• Práctica de desarrollo de código *test-first* con *mocking*(2 horas)	
Semana 12 (7 horas)	*Hamcrest matchers* (2 horas)		• Estudio individual (2 horas) • Resolución de ejercicios y problemas (3 horas)			
Semana 13 (8 horas)	Extensión de *frameworks* de testing (2 horas)		• Estudio individual (2 horas) Resolución de ejercicios y problemas (4 horas)			
Semana 14 (6 horas)					• Práctica de adaptación a las necesidades del programador de un *framework* de testing (6 horas)	
Semana 15 (2 horas)	Recapitulación y discusión (2 horas)					

Sistema de evaluación de la asignatura

EVALUACION		
Ref	**INDICADOR DE LOGRO**	**Relacionado con RA:**
I1	Desarrollo de código siguiendo la estrategia *test-last*	RA1, R3
I2	Desarrollo de código siguiendo la estrategia *incremental test-last*	RA1, RA2, RA3
I3	Desarrollo de código siguiendo la estrategia *test-first,* sin uso de *mocking*	RA2, RA3
I4	Desarrollo de código siguiendo la estrategia *test-first,* con uso de *mocking*	RA2, RA3
I5	Adaptación a las necesidades del programador de un *framework* de testing	R3

La tabla anterior puede ser sustituida por la tabla de rúbricas.

EVALUACION SUMATIVA			
BREVE DESCRIPCION DE LAS ACTIVIDADES EVALUABLES	**MOMENTO**	**LUGAR**	**PESO EN LA CALIFICACIÓN**
Desarrollo de código siguiendo la estrategia *test-last*	Semanas 2-4	Aula Informática	20%
Desarrollo de código siguiendo la estrategia *incremental test-last*	Semana 5	Aula Informática	20%
Desarrollo de código siguiendo la estrategia *test-first,* sin uso de *mocking*	Semana 8	Aula Informática	20%
Desarrollo de código siguiendo la estrategia *test-first,* con uso de *mocking*	Semana 9	Aula Informática	20%
Adaptación a las necesidades del programador de un *framework* de testing	Semana 10	Aula Informática y casa	20%
Rendimiento en clase	Semanas 1-15	Clase	10%

CRITERIOS DE CALIFICACIÓN

Los alumnos deben aprobar (esto es, obtener al menos un 5.0) en tres de las cinco prácticas del curso. La nota final se calculará como la media aritmética de las notas obtenidas en las prácticas, a lo que se añadirá el rendimiento en clase.

Los alumnos tendrán la opción de repetir una y sólo una de las cinco prácticas (por ejemplo, para subir nota).

Los alumnos que deseen ser evaluados por examen final, deberán comunicárselo por escrito (ej: email) a los profesores durante el mes de Septiembre.