GUIA DOCENTE DE LA ASIGNATURA

NUEVAS TECNOLOGÍAS DE LA PROGRAMACIÓN

MÓDULO	MATERIA	CURSO	SEMESTRE	CRÉDITOS	TIPO
COMPLEMENTOS DE INGENIERÍA DEL SOFTWARE	COMPLEMENTOS DE PROGRAMACIÓN	4	2	6	Optativa

PROFESOR(ES)	DIRECCIÓN COMPLETA DE CONTACTO PARA TUTORÍAS (Dirección postal, teléfono, correo electrónico, etc.)
Manuel Gómez Olmedo	Despacho 31, Departamento de Ciencias de la Computación e Inteligencia Artificial ETS. Ingenierías Informática y de Telecomunicaciones. C/ Periodista Daniel Saucedo Aranda, s/n Granada 18001 Teléfono: 951246143 Correo electrónico: Manuel Gómez Olmedo: mgomez@decsai.ugr.es
	HORARIO DE TUTORÍAS Disponibles en: http://decsai.ugr.es/index.php?p=profesores&id=9847

GRADO EN EL QUE SE IMPARTE	OTROS GRADOS A LOS QUE SE PODRÍA OFERTAR
Grado en Ingeniería Informática	

PRERREQUISITOS Y/O RECOMENDACIONES (Si ha lugar)

Los alumnos no habrán de tener materias o asignaturas aprobadas como requisito indispensable para superar esta materia. No obstante, se recomienda tener aprobados los contenidos y adquiridas las competencias de cuatrimestres precedentes.

BREVE DESCRIPCIÓN DE CONTENIDOS (SEGÚN MEMORIA DE VERIFICACIÓN DEL GRADO)

Técnicas avanzadas de programación: metadatos y reflexión. Tipos genéricos. Patrones y antipatrones. Aplicaciones.

COMPETENCIAS GENERALES Y ESPECÍFICAS

Competencias Generales del Título

E9. 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.

Competencias Básicas

CB5. Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía.

OBJETIVOS (EXPRESADOS COMO RESULTADOS DE APRENDIZAJE)

Objetivos formativos particulares

- Comprender la necesidad de almacenar información sobre las clases usadas en una aplicación orientada a objetos, así como las técnicas necesarias para llevarlo a cabo.
- Conocer la forma en que se recupera y usa, en tiempo de ejecución, la información sobre el contenido interno de las clases usadas, sus datos y métodos miembro.
- Conocer algunos detalles sobre la forma en que se usa la reflexión en diferentes lenguajes de programación.
- Conocer las ventajas obtenidas al parametrizar las clases, usando el mecanismo ofrecido por los tipos genéricos.
- Comprender las ventajas y limitaciones impuestas por el uso de los tipos genéricos.
- Practicar con el uso de tipos genéricos en diferentes lenguajes de programación.
- Analizar de forma crítica un diseño software, identificando puntos críticos que supongan un problema en cuanto a su posible ampliación o modificación.
- Conocer soluciones prototípicas que han demostrado su eficacia en la resolución de problemas concretos pero usualmente presentes en los sistemas software.
- Ser capaces de identificar qué patrón de diseño puede solucionar un determinado problema identificado en un diseño software.
- Conocer los antipatrones e identificarlos en diseños propuestos para su estudio.
- Adquirir capacidades en el uso avanzado de lenguajes de programación, no cubiertas en otras materias y asignaturas, como, por ejemplo, el paradigma de programación funcional.

Objetivos formativos de carácter general (Competencias según BOE de 4 de Agosto de 2009)

Ser capaz de identificar y analizar problemas y diseñar, desarrollar, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales.

TEMARIO DETALLADO DE LA ASIGNATURA

TEMA 1. Repaso de conceptos de orientación a objetos.

Conceptos de clase y objeto, herencia, polimorfismo, uso de interfaces.

TEMA 2. Patrones y antipatrones. Refactorización.

Introducción. Ejemplos motivadores y aplicaciones de patrones: observador, decorador, factoría, factoría abstracta, singleton, plantilla, iterador, composite, estado. Introducción a la refactorización.

TEMA 3. Java8: programación funcional en Java

Introducción. Uso de colecciones. String, comparadores y filtros. Diseño con expresiones lambda. Trabajo con recursos. Evaluación lazy. Optimización de recursos. Composición con expresiones lambda. Recapitulación.

TEMA 4. Programación en Scala: paradigma de programación funcional

Introducción. Variables, estructuras de control, funciones, clases y objetos, herencia.

TEMARIO DE PRÁCTICAS

PRÁCTICA 1. Entorno de trabajo, repaso de conceptos previos de orientación a objetos.

PRÁCTICAS 2. Práctica sobre programación funcional en Java.

PRÁCTICA 3. Práctica inicial de Scala: primeros pasos.

PRÁCTICA 4. Práctica sobre funciones y cierres.

PRÁCTICA 4. Práctica sobre composición y herencia.

PRÁCTICA 5. Trabajo de evaluación de la asignatura.

SEMINARIOS

SEMINARIO 1. Importancia del diseño de sistemas software.

SEMINARIO 2. Producción de software en el mundo real.

BIBLIOGRAFÍA

FUNDAMENTAL:

- Head First Design Patterns. E. Freeman, E. Freeman, B. Bates, K. Sierra. O'Reilly Media, 2004.
- Design Patterns Explained. A. Shalloway, J.R. Trott. Addison-Wesley, 2004.
- Programming in Scala, M. Odersky, L. Spoon, B. Venners. Artima, 2014.
- Anti Patterns. Refactoring software, Architectures and Projects in Crisis. W.H. Brown, R.C. Malvean, H.W. McCormick, T.J. Mowbray. Wiley, 1998.Java Reflection in Action. I.R. Forman, N. Forman. Manning Publications, 2004.
- J2EE Antipatterns. B. Dudney, S. Asbury, J. Krozak, K. Wittkopf. Wiley, 2003.
- Introduction to Java Programming. Y.D. Liang. Prentice Hall, 2014.
- Big Java: early objects. C. Horstman. Wiley, 2013.
- Java Generics and Collections. N. Naftalin, P. Wadler. O'Reilly Media, 2006.
- Object-Oriented Reengineering Patterns. S. Demeyer, S. Ducase, O. Nierstrasz. Morgan-Kaufman, 2002.
- Refactoring: improving the design of existing code. M. Fowler, K. Beck, J. Brant, W. Opdyke, D. Roberts. Addison-Wesley Professional, 1999.
- Functional Programming in Java: harnessing the power of Java 8 lambda expressions. V. Subramaniam. The pragmatic programmers, 2014.

BIBLIOGRAFÍA COMPLEMENTARIA:

- Head First Java (2ª edición). B. Bates. O'Reilly Media, 2009.
- Thinking in Java (4ª edición). B. Eckel. Prentice Hall PTR, 2006.
- Core Java, vols I y II. C.S. Horstmann, G. Cornell. Prentice Hall PTR, 2008.
- Java in a Nutshell (5 ª edición). D. Flanagan. O'Reilly Media, 2005.

ENLACES RECOMENDADOS

Java design patterns at a glance: http://www.javacamp.org/designPattern/

Java programming design patterns: http://en.wikibooks.org/wiki/Java_Programming/Design_Patterns

Refactoring: http://refactoring.com/

Design patterns, antipatterns, refactoring: http://sourcemaking.com/

Functional programming in Java: http://blog.jetbrains.com/blog/2014/03/27/functional-programming-with-java-8/

METODOLOGÍA DOCENTE

1. **Lección magistral (Clases teóricas-expositivas) (grupo grande)**
Descripción: Presentación en el aula de los conceptos propios de la materia haciendo uso de metodología expositiva con lecciones magistrales participativas y medios audiovisuales. Evaluación y examen de las capacidades adquiridas.
Propósito: Transmitir los contenidos de la materia motivando al alumnado a la reflexión, facilitándole el descubrimiento de las relaciones entre diversos conceptos y formarle una mentalidad crítica
Contenido en ECTS: 7.2
Competencias: CB5, E9

2. **Actividades prácticas (Clases prácticas de laboratorio) (grupo pequeño)**
Descripción: Actividades a través de las cuales se pretende mostrar al alumnado cómo debe actuar a partir de la aplicación de los conocimientos adquiridos
Propósito: Desarrollo en el alumnado de las habilidades instrumentales de la materia.
Contenido en ECTS: 3.6
Competencias: CB5, E9

3. **Seminarios (grupo pequeño)**
Descripción: Modalidad organizativa de los procesos de enseñanza y aprendizaje donde tratar en profundidad una temática relacionada con la materia. Incorpora actividades basadas en la indagación, el debate, la reflexión y el intercambio.
Propósito: Desarrollo en el alumnado de las competencias cognitivas y procedimentales de la materia.
Contenido en ECTS: 2.4
Competencias: CB5, E9

4. **Actividades no presenciales individuales (Estudio y trabajo autónomo)**
Descripción: 1) Actividades (guiadas y no guiadas) propuestas por el profesor a través de las cuales y de forma

individual se profundiza en aspectos concretos de la materia posibilitando al estudiante avanzar en la adquisición de determinados conocimientos y procedimientos de la materia, 2) Estudio individualizado de los contenidos de la materia 3) Actividades evaluativas (informes, exámenes, ...)
Propósito: Favorecer en el estudiante la capacidad para autorregular su aprendizaje, planificándolo, diseñándolo, evaluándolo y adecuándolo a sus especiales condiciones e intereses.
Contenido en ECTS:10.8
Competencias: CB5, E9

5. **Actividades no presenciales grupales (Estudio y trabajo en grupo)**
Descripción: Actividades (guiadas y no guiadas) propuestas por el profesor a través de las cuales y de forma grupal se profundiza en aspectos concretos de la materia posibilitando a los estudiantes avanzar en la adquisición de determinados conocimientos y procedimientos de la materia.
Propósito: Favorecer en los estudiantes la generación e intercambio de ideas, la identificación y análisis de diferentes puntos de vista sobre una temática, la generalización o transferencia de conocimiento y la valoración crítica del mismo.
Contenido en ECTS: 10.8
Competencias: CB5, E9

6. **Tutorías académicas (grupo pequeño)**
Descripción: manera de organizar los procesos de enseñanza y aprendizaje que se basa en la interacción directa entre el estudiante y el profesor
Propósito: 1) Orientan el trabajo autónomo y grupal del alumnado, 2) profundizar en distintos aspectos de la materia y 3) orientar la formación académica-integral del estudiante
Contenido en ECTS: 1.2
Competencias: CB5, E9

EVALUACIÓN (INSTRUMENTOS DE EVALUACIÓN, CRITERIOS DE EVALUACIÓN Y PORCENTAJE SOBRE LA CALIFICACIÓN FINAL, ETC.)

La adaptación del sistema de evaluación continua general propuesto a las características de esta asignatura, con indicación explícita del peso de la evaluación de cada actividad formativa, se ajustará a lo indicado en la siguiente tabla:

Actividades Formativas	Ponderación
Parte Práctica: proyecto final de la asignatura	35.00%
Parte Práctica: prácticas en aula, prácticas evaluables	55.00%
Evaluación continua	10.00%

L a **evaluación única final** y **la evaluación en convocatorias extraordinarias** se realizará en un solo acto académico el día de la convocatoria oficial de examen para la asignatura. Dicha prueba (evaluada de 0 a 10) incluirá preguntas tanto de tipo teórico como práctico que garanticen que el alumno ha adquirido la totalidad de las competencias descritas en esta misma guía docente.

De acuerdo a lo establecido en la Normativa de evaluación y de calificación de los estudiantes de la Universidad de Granada aprobada en Consejo de Gobierno de 20 de mayo de 2013 (NCG71/2), la evaluación será preferentemente continua. No obstante, el estudiante que no pueda acogerse a dicho sistema por motivos laborales, estado de salud, discapacidad o cualquier otra causa debidamente justificada podrá acogerse a la evaluación única final. Para ello deberá solicitarlo al Director del Departamento en las dos primeras semanas de impartición de la asignatura o, excepcionalmente, en las dos primeras semanas tras la matriculación en la asignatura (NCG78/9: Instrucción relativa a la aplicación del artículo 8.2).

RÉGIMEN DE ASISTENCIA
La asistencia a las clases teóricas y prácticas no será obligatoria, aunque la participación activa en clase y la entrega de ejercicios planteados por el profesor se tendrá en cuenta dentro del sistema de evaluación continua de la asignatura. Sí será obligatoria la defensa de prácticas si así se requiere por parte del profesor (en caso de no realizarse la defensa la práctica correspondiente se considerará como no entregada).

INFORMACIÓN ADICIONAL
Definición de grupo grande y grupo pequeño: Los grupos grandes son grupos de 45 a 60 estudiantes. Los grupos pequeños son grupos de 15 a 20 estudiantes.