Universidad de Burgos

Programa de la asignatura **Curso: 2009 / 2010**

METODOLOGÍA DE LA PROGRAMACIÓN (1298)

PROFESORADO

Profesor/es:
RAUL MARTICORENA SANCHEZ - correo-e: rmartico@ubu.es
DAVID HERMINDO MARTIN ALONSO - correo-e: dmartin@ubu.es

FICHA TÉCNICA

Titulación: INGENIERÍA TÉCNICA EN INFORMÁTICA DE GESTIÓN
Centro: ESCUELA POLITÉCNICA SUPERIOR
Nombre asignatura: METODOLOGÍA DE LA PROGRAMACIÓN (1298)
Código de la asignatura: 1298
Tipo de asignatura: Troncal
Nivel / Ciclo: 1
Curso en el que se imparte: 2
Duración y fechas: Cuatrimestral - 1er Cuatrimestre
Créditos: 6.0
Créditos teóricos: 3.0
Créditos prácticos: 3.0
Áreas: LENGUAJES Y SISTEMAS INFORMATICOS
Tipo de curso: Oficial
Descriptores: Según BOE
Requisitos previos: Según BOE
Idioma: Español

COMPETENCIAS TRANSVERSALES O GENÉRICAS

INSTRUMENTALES

Análisis y síntesis: 4
Organización y planificación: 4
Comunicación oral y escrita en la lengua nativa: 4
Conocimiento de una lengua extranjera: 3
Conocimientos de informática relativos al ámbito de estudio: 4
Gestión de la información: 3
Resolución de problemas: 4
Toma de decisiones: 4

PERSONALES

Trabajo en equipo: 3
Trabajo en un equipo de carácter interdisciplinar: 3

Trabajo en un contexto internacional: 1
Relaciones interpersonales: 3
Reconocimiento a la diversidad y la multiculturalidad: 3
Razonamiento crítico: 4
Compromiso ético: 3

SISTÉMICAS

Aprendizaje autónomo: 3
Adaptación a nuevas situaciones: 4
Creatividad: 4
Liderazgo: 3
Conocimiento de otras culturas y costumbres: 1
Iniciativa y espíritu emprendedor: 4
Motivación por la calidad: 4
Sensibilidad hacia temas medioambientales: 1

COMPETENCIAS ESPECÍFICAS

CONOCIMIENTOS DISCIPLINARES (SABER)

Diseño de programas: descomposición modular y documentación.

Programación orientada a objetos. Mecanismos básicos: clases, objetos. Conceptos avanzados: herencia, genericidad y excepciones. Diseño de programas siguiendo este paradigma.

Técnicas de verificación y prueba de programas.

Convenciones y reglas en la documentación de programas.

HABILIDADES PROFESIONALES (SABER HACER)

Implementar un conjunto de clases a partir de un diseño planteado sobre un supuesto práctico.

Aplicar las convenciones de estilo de un lenguaje orientado a objetos, documentando adecuadamente el código.

Determinar los casos de prueba a aplicar a una clase y programarlos con un framework de pruebas unitarias.

ACTITUDES (SABER SER - SABER ESTAR)
COMP. ACADÉMICAS (SABER TRASCENDER)
OTRAS COMPETENCIAS ESPECÍFICAS

Desarrollar hábitos de programación orientados a conseguir un software de calidad.

Comunicar y expresar de forma adecuada los elementos y circunstancias que rodean el proceso de desarrollo, especialmente en el contexto referido, de programación orientada a objetos.

OTROS OBJETIVOS DE LA ASIGNATURA

METODOLOGÍA Y RECURSOS PARA EL APRENDIZAJE

Clases magistrales de teoría apoyadas en transparencias junto con la realización de guiones de prácticas en laboratorio. Se solicitan dos trabajos prácticos obligatorios.

BREVE DESCRIPCIÓN DE LAS ACTIVIDADES PRÁCTICAS

Se realizan en grupos de dos personas máximo. Se propondrán dos prácticas: un sistema programado con Java siguiendo el paradigma de orientación a objetos y una batería de pruebas realizadas con un framework de pruebas automáticas. Los alumnos deben entregarlas dentro de las fechas señaladas. Si no se presentan las dos prácticas en las fechas marcadas supone una calificación cero (PR=0). En la convocatoria extraordinaria se propondrá un nuevo enunciado de prácticas.

SEGUIMIENTO DEL ALUMNO Y CRITERIOS DE EVALUACIÓN

Se evaluará mediante:
-Examen escrito que consistirá en:
-preguntas teóricas y problemas (ET) : 5 puntos (nota mínima 2.25 puntos)
-preguntas relacionadas con prácticas de laboratorio (EP) : 2.5 puntos (nota mínima 1 punto)
- Prácticas obligatorias (PR): 2.5 puntos (nota mínima 1.25 puntos)
La nota final, una vez superadas las notas mínimas, se obtiene mediante la suma de las tres notas y debe igualar o superar los 5 puntos:
-ET + EP + PR >= 5

Si no se cumplen las condiciones anteriores para aprobar la asignatura en la convocatoria ordinaria, se guardará cualquiera de las tres notas que superen su nota mínima (ET, EP o PR) para la convocatoria extraordinaria pero no para cursos sucesivos. En caso de copia en examen o prácticas obligatorias, no se guardará ninguna parte con nota mínima, al margen de cualquier otra sanción disciplinaria que puedan recoger los reglamentos aplicables.

BIBLIOGRAFÍA BÁSICA SOBRE LA MATERIA

Construcción de Software Orientado a Objetos, *Bertrand Meyer*, 2ªEdición, 1998, Prentice Hall,
El arte de probar el software, *Myers, G.J*, 1ª Edición, 1984, El Ateneo,
El lenguaje de Programación Java, *Arnold, K. , Gosling, J., Holmes, D.*, 3ª Edición, 2001, Addison-Wesley,
Matemática discreta y lógica, *Grassmann, W.K and Tremblay, J.P*, 1ª Edición, 1997, Prentice Hall.,
Piensa en Java, *Bruce Eckel*, 2ªEdición, 2001, Prentice Hall,
Programación en Java 2. Serie Schaum, *Sánchez Allende, Jesús y otros*, 1ª Edición, 2005,

McGraw-Hill, España

BIBLIOGRAFÍA COMPLEMENTARIA

Design by Contract, by Example, *Richard Mitchell and Jim McKin*, 1st Edition, 2002, Addison-Wesley,
Ingeniería del Software. Un enfoque práctico, *Pressman, R.S*, 5ª Edición, 2001, Mc Graw-Hill,
Principles of Object-Oriented Software Development, *Eliëns, A.*, 1st Edition, 2000, Addison-Wesley,
Program Development in Java: Abstraction, Specification and Object-Oriented Design, *Barbara Liskov and John Guttag*, 1st Edition, 2000, Addison-Wesley,
Testing Object-Oriented System. Models, Patterns and Tools, *Binder, R.*, 1st Edition, 2000, Addison-Wesley,

RECURSOS DE INTERNET

OBSERVACIONES Y OTROS DATOS

ESTRUCTURA DE CONTENIDOS (TEMAS)

METODOLOGÍA DE LA PROGRAMACIÓN (1298)

Unidad A. Programación Orientada a Objetos

> Tema 1. Introducción
- 1. Modularidad: criterios, reglas y principios
- 2. Reusabilidad
- 3. Descomposición: funcional y basada en objetos
- 4. Tipos abstractos de datos
- 4. Tipos abstractos de datos

> Tema 2. La estructura estática: Clases
- 1. El papel de las clases
- 2. Sistema de tipos uniforme
- 3. Definición de clases
- 4. Paso de mensajes
- 5. Sistemas
- 6. Ocultación de la información
- 7. Clases en Java

> Tema 3. La estructura dinámica: Objetos
- 1. Objetos: herramienta de modelado
- 2. Manipulación de objetos y referencias
- 3. Creación de objetos
- 4. Operaciones sobre referencias
- 5. Objetos compuestos y tipos expandidos.
- 6. Uso de referencias
- 7. Gestión de memoria
- 8. Objetos en Java

> Tema 4. Herencia
- 1. Extensión de clase por herencia
- 2. Polimorfismo
- 3. Tipos y herencia
- 4. Ligadura dinámica
- 5. Características y clases diferidas (clases abstractas)
- 6. Redeclaración
- 7. Herencia múltiple y herencia repetida
- 8. Herencia en Java

> Tema 5. Genericidad.
- 1. Generalización de tipos.
- 2. Parametrización de tipos
- 3. Clases genéricas
- 4. Ejemplos de genericidad
- 5. Coste de la genericidad
- 6. Genericidad en Java

> Tema 6. Programación Defensiva y Tratamiento de Excepciones
- 1. Concepto de programación defensiva
- 2. Manejo de excepciones

- 3. Excepciones en Java

Unidad B. Corrección del Software

> Tema 7. Pruebas del Software
  - 1. Psicología de las pruebas
  - 2. Pruebas de caja negra
  - 3. Pruebas de caja blanca
  - 4. Prueba en OO
  - 5. Pruebas unitarias y de integración
  - 6. Herramientas para pruebas
  - 7. Depuración
  - 8. Pruebas en Java

> Tema 8. Verificación Formal
  - 1. Conceptos preliminares
  - 2. Programas sin bucles
  - 3. Programas con bucles
  - 4. Programas con vectores

> Tema 9. Diseño por Contrato
  - 1. Introducción
  - 2. Precondiciones y postcondiciones
  - 3. Invariantes de clase
  - 4. Aserciones en el código.
  - 5. Bucles: variantes e invariantes
  - 6. Herencia y aserciones
  - 7. Utilización de aserciones
  - 8. Aserciones en Java