UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA

GUÍA DOCENTE *CURSO: 2013/14*

40813 - PROGRAMACIÓN I

CENTRO: *180 - Escuela de Ingeniería Informática*
TITULACIÓN: *4008 - Grado en Ingeniería Informática*
ASIGNATURA: *40813 - PROGRAMACIÓN I*

Vinculado a : (Titulación - Asignatura - Especialidad)
4801-Doble Grado en Ingeniería Informática y - 48113-PROGRAMACIÓN I - 00

CÓDIGO UNESCO: *1203* **TIPO:** *Obligatoria* **CURSO:** *2* **SEMESTRE:** *1º semestre*
CRÉDITOS ECTS: *6* **Especificar créditos de cada lengua:** **ESPAÑOL:** *6* **INGLÉS:**

SUMMARY

REQUISITOS PREVIOS

- Introducción a la Informática
- Fundamentos de Programación

Datos identificativos del profesorado que la imparte.

Plan de Enseñanza (Plan de trabajo del profesorado)

Contribución de la asignatura al perfil profesional:

Programación I es una asignatura que aporta al estudiante una formación básica en el modelo de programación orientado a objetos y en el de programación guiada por eventos. La asignatura emplea un lenguaje de programación que ilustra de forma sencilla los principales elementos de ambos paradigmas y que facilita el adiestramiento en el desarrollo de interfaces gráficas de usuario. Además, se introducen los conceptos de tipos abstractos de datos y contenedor cuya concreción en las colecciones ofertadas por el lenguaje proporciona elementos muy útiles para la resolución de gran número de problemas. Asimismo, se inicia al estudiante en las técnicas de prueba de programas utilizando paquetes de apoyo específicos.

Competencias que tiene asignadas:

Competencias generales:
G1- Poseer y comprender conocimientos en un área de estudio (Ingeniería Informática) que parte de la base de la educación secundaria general, y se suele encontrar a un nivel que, si bien se apoya en libros de texto avanzados, incluye también algunos aspectos que implican conocimientos procedentes de la vanguardia de su campo de estudio.
G2- Aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio.
G3- Reunir e interpretar datos relevantes (normalmente dentro de su área de estudio) para emitir

juicios que incluyan una reflexión sobre temas relevantes de índole social, científica o ética.
G5- Desarrollar aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía.

Competencias nucleares de la Universidad de Las Palmas de Gran Canaria:
N4- Comprometerse activamente en el desarrollo de prácticas profesionales respetuosas con los derechos humanos así como con las normas éticas propias de su ámbito profesional para generar confianza en los beneficiarios de su profesión y obtener la legitimidad y la autoridad que la sociedad le reconoce.

Competencias del título de Grado en Ingeniería Informática:
T3. 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. (G1, G2)
T7. Capacidad para conocer, comprender y aplicar la legislación necesaria durante el desarrollo de la profesión de Ingeniero Técnico en Informática y manejar especificaciones, reglamentos y normas de obligado cumplimiento. (N4)
T8. 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. (G3, N3)
T9. 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. (G4, N1)

Competencias comunes a la Ingeniería Informática:
CII05. Conocimiento, administración y mantenimiento sistemas, servicios y aplicaciones informáticas.
CII08. Capacidad para analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados.

Objetivos:

1) Aplicar el paradigma de programación orientada a objetos en el desarrollo de programas y usar herramientas específicas de apoyo.
2) Probar programas utilizando pruebas de unidades
3) Utilizar contenedores de forma apropiada en el desarrollo de programas.
4) Aplicar el modelo de programación guiada por eventos de forma adecuada en el desarrollo de programas.

Contenidos:

1. Programación Orientada a Objetos
  1.1. Introducción a la Programación Orientada a Objetos
  1.2. Clases, objetos
  1.3. Modularidad
  1.4. Herencia
  1.5. Polimorfismo
  1.6. Clases abstractas e interfaces
  1.7. Introducción a UML
Competencias: G1, G2l G3, G4, G5, N4, T3, T7, T8, T9, CII05
Bibliografía: [1] [2] [3] [5]

2. Robustez y prueba de Programas

2.1. Aserciones
2.2. Excepciones
2.3. Técnicas de prueba de programas
2.4. Diseño de casos de prueba
2.5. Pruebas unitarias
Competencias: G1, G2l G3, G4, G5, N4, T3, T7, T8, T9, CII05, CII08
Bibliografía: [1] [4] [6] [7]

3. Genericidad
3.1. Clases genéricas
3.2. Métodos genéricos
Competencias: G1, G2l G3, G4, G5, N4, T3, T7, T8, T9, CII05
Bibliografía: [3]

4. Contenedores
4.1. Introducción
4.2. Clases contenedoras
Competencias: G1, G2l G3, G4, G5, N4, T3, T7, T8, T9, CII05, CII08
Bibliografía: [3]

5. Fundamentos de Programación Guiada por Eventos
5.1. Introducción
5.1. Gestores de eventos
5.2. Interfaz gráfica de usuarios
Competencias: G1, G2l G3, G4, G5, N4, T3, T7, T8, T9, CII05, CII08
Bibliografía: [3]

Metodología:

• Sesiones académicas teóricas. En ellas el profesor expondrá aspectos teóricos relevantes de los contenidos de la asignatura dando respuesta colectiva a las dudas que puedan surgir durante el estudio personal de estos.

• Sesiones académicas prácticas. En ellas el profesor expondrá las actividades prácticas a realizar, mostrando el uso de las herramientas necesarias para su realización y guiando en las dudas que plantee su desarrollo.
Las prácticas a realizar serán:
- Clases, métodos y constructores
- Herencia y Polimorfismo
- Prueba de programas
- Contenedores
- Interfaz gráfica de usuarios

• Ejercicios de autoevaluación. Se pondrá a disposición de los estudiantes ejercicios que permitan una autoevalaución sencilla de las habilidades adquiridas.

• Tutorías colectivas. Se usarán en las actividad en grupo para resolver dudas y hacer un seguimiento de éstas.

• Tutorías individuales. Se usarán para resolver dudas en general y hacer un seguimiento de las actividad individuales.

Criterios y fuentes para la evaluacion:

Fuentes para la evaluación:
1) Exámenes.
2) Trabajos prácticos en equipo.
3) Coevaluación.

1) Exámenes
Criterios relativos a las cuestiones:
-Conocimiento y comprensión de los contenidos
-Rigor y fluidez en la comunicación escrita
Criterios relativos al desarrollo de un programa:
-El profesor someterá el programa a las pruebas oportunas para formular una valoración que tenga en cuenta todos los aspectos implicados en su realización, con especial énfasis en la ejecución (funcionamiento, adecuación a las especificaciones, robustez, ...) y en el estilo (formato, comentarios, elección de identificadores, ...).
Las competencias G1, G2, T3, T7, T8, T9, CII05 y CII08 se evaluarán parcial o totalmente mediante los exámenes.

2) Trabajos prácticos en equipo
Criterios relativos a una práctica:
-El profesor someterá los programas e información entregados a los análisis y pruebas oportunos para formular una valoración global que tenga en cuenta todos los aspectos implicados en su realización, con especial énfasis en la ejecución (funcionamiento, adecuación a las especificaciones, robustez, ...), en el estilo (formato, comentarios, elección de identificadores, ...). Cuando lo estime conveniente, el profesor podrá citar a los alumnos para formularles cuestiones que considere relevantes para la valoración global.
Las competencias G1, G2, N4, T7, T8, T9, CII05 y CII08 se evaluarán parcial o totalmente mediante la valoración de los trabajos prácticos de programación en equipo.

3) Coevaluación
Cada miembro del equipo valora a sus compañeros teniendo en cuenta su grado de: colaboración, respeto, integración, distribución del trabajo, motivación, compromiso y conciliación. El profesor supervisará la apropiada y justa evaluación entre los miembros de los equipos.
Las competencias G1, G3, G5, N4, T8 y CII01 se evaluarán parcial o totalmente mediante esta fuente.

Sistemas de evaluacion:

1) Evaluación continua:
* Exámenes individuales
La escala que se utilizará para valorar esta parte es de 0 a 10 puntos y se efectuará a partir de exámenes individuales que se irán realizando regularmente, este sistema se basa en cuestiones de respuesta corta y/o desarrollo de programas relativos a los temas tratados. Cada examen aporta el mismo porcentaje a la nota de esta parte y se valora de 0 a 10 puntos.

* Trabajo práctico en equipo
La escala que se utilizará para valorar el trabajo práctico es de 0 a 10 puntos y su evaluación se efectuará a partir de las prácticas de laboratorio. El enunciado de cada práctica incluirá las especificaciones, plazos de entrega y porcentaje que aporta a la evaluación del trabajo práctico. Cada equipo deberá entregar, dentro de los plazos establecidos, los ficheros conteniendo el código fuente y demás información requerida, usando el medio que se establezca. La nota obtenida en cada práctica es la que le corresponderá a cada alumno componente del equipo que la realiza.

* Coevaluación
Cada alumno interviene en la evaluación al calificar la participación activa en el trabajo colaborativo de sus compañeros de equipo por medio de un cuestionario que se valorará de 0 a 10 puntos. La nota de un alumno por cada coevaluación realizada será la media de la otorgada por sus compañeros de equipo. La nota final de esta parte será la media de los resultados obtenidos en las coevaluaciones efectuadas a lo largo del semestre y se valorará de 0 a 10 puntos. Requiere la asistencia a las sesiones académicas, con un mínimo del 80%. Los resultados de la coevalaución serán supervisados por el profesor para evitar posibles disfunciones en la nota.

2) Examen escrito final
Aquellos alumnos que no superen la evaluación continua podrán optar a un examen escrito. El examen estará dividido en cuestiones de respuesta corta y/o desarrollo de programas, cada uno de los cuales aportará un porcentaje especificado a la nota. Se valorará de 0 a 10 puntos.

Criterios de calificacion:

Para superar la asignatura habrá que obtener una nota mínima de 5 puntos en la evaluación continua de exámenes individuales. La nota final se calcula con la siguiente fórmula:

NF = NEI * 0.5 + NTP * 0.3 + NC * 0.2

Donde NF representa la nota final, NEI la nota de exámenes individuales, NTP la nota del trabajo práctico en equipo y NC la nota de la coevaluación. NEI será sustituida en la fórmula por la nota obtenida mediante el examen escrito final para aquellos alumnos que no alcancen la nota mínima establecida para la evaluación continua de exámenes individuales. La nota final de los alumnos que no cumplan las condiciones establecidas para superar la asignatura será como máximo de 4 puntos.

Plan de Aprendizaje (Plan de trabajo de cada estudiante)

Tareas y actividades que realizará según distintos contextos profesionales (científico, profesional, institucional, social)

1) Búsqueda, consulta y estudio de documentación e información
2) Actividades formativas consistentes en la realización de programas en equipo que requieren del desarrollo individual para una posterior discusión colectiva.
3) Trabajo práctico consistente en la realización de las prácticas de laboratorio en equipo que requieren del desarrollo individual para una posterior discusión colectiva.
4) Uso del entorno de desarrollo integrado. Desarrollo de aplicaciones con interfaz gráfica. Prueba de programas empleando pruebas unitarias.

Científico: 1, 2, 3, 4
Profesional: 1, 2, 3, 4
Institucional:
Social: 2, 3

Temporalización semanal de tareas y actividades (distribución de tiempos en distintas actividades y en presencialidad - no presencialidad)

Los estudiantes dedicarán 2 horas semanales a asistir a sesiones académicas teóricas y otras dos a sesiones académicas prácticas, que incluirán la realización de ejercicios de aplicación de las

técnicas introducidas en la teoría y la solución de problemas de programación (60 horas en 15 semanas). Los exámenes que se realizarán a lo largo del semestre consumen 7.5 horas presenciales en total.
En promedio, realizarán cada semana 5 horas de trabajo no presencial: 2 horas dedicadas a la consulta y asimilación de materiales documentales y 3 horas para la realización de ejercicios y trabajos; ello supone 75 horas en 15 semanas.
Previsión de dedicación temporal en la relación de prácticas y actividades formativas.
De la semana 1 a la 4 se realizarán actividades formativas prácticas no evaluables. De la semana 5 a la 15 se realizarán 5 prácticas que constarán de dos semanas (una inicial de actividad individual y otra de tarea en equipo).
Las horas restantes las distribuirá el estudiante, en función de sus necesidades particulares, entre atención tutorial individual, tiempo adicional de estudio y realización de ejercicios y trabajos.

Recursos que tendrá que utilizar adecuadamente en cada uno de los contextos profesionales.

1) Bibliografía, documentación específica sobre el contenido de la asignatura y documentación técnica.
2) Herramientas de desarrollo (Java, interfaz gráfica en Java, prueba de programas, diseño con UML)
3) Navegador web (Herramientas del campus virtual de la ULPGC, herramienta de desarrollo online)

CONTEXTOS

Científico: 1, 2, 3
Profesional: 1, 2, 3
Institucional:
Social: 3

Resultados de aprendizaje que tendrá que alcanzar al finalizar las distintas tareas.

• Explicar el concepto, implementar y usar de forma adecuada en el desarrollo de programas los tipos abstractos de datos.
• Describir los elementos básicos de la programación orientada a objetos y aplicarlos de forma adecuada en el desarrollo de programas.
• Explicar los conceptos de contenedor e iterador y usarlos de forma adecuada en el desarrollo de programas.
• Emplear técnicas de prueba para detectar errores en programas de complejidad media.
• Nombrar y usar librerías estándar de clases.
• Explicar los fundamentos de la programación guiada por eventos y usarla en el desarrollo de programas simples.

Plan Tutorial

Atención presencial individualizada (incluir las acciones dirigidas a estudiantes en 5ª, 6ª y 7ª convocatoria)

La atención presencial individualizada se verificará en el horario de tutorías establecido para los profesores de la asignatura y publicado en la web del Departamento de Informática y Sistemas.

Atención presencial a grupos de trabajo

La atención presencial a grupos de trabajo se verificará en el horario de tutorías establecido para los profesores de la asignatura y publicado en la web del Departamento de Informática y Sistemas.

Atención telefónica

La atención teléfonica se verificará en los teléfonos del profesorado publicados en la web del Departamento de Informática y Sistemas, en el horario de tutorías establecido para los profesores de la asignatura y publicado en la web del Departamento de Informática y Sistemas.

Atención virtual (on-line)

La atención virtual (on-line) se llevará a cabo usando las herramientas disponibles en el Campus Virtual de la ULPGC. Se responderá a las consultas virtuales en el horario de tutorías establecido para los profesores de la asignatura y publicado en la web del Departamento de Informática y Sistemas.

Bibliografía

[1 Básico] Programación orientada a objetos con Java /

Francisco Durán, Francisco Gutiérrez, Ernesto Pimentel.
Thomson Paraninfo,, Madrid : (2007)
9788497325721

[2 Básico] UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado /

Larman Craig.
Prentice Hall,, Madrid : (2003) - (2ª ed.)
8420534382

[3 Básico] The Java tutorials [

Oracle and /or its affiliates.
Oracle,, [s.l.] : (1995)

[4 Básico] Java tools for Extreme Programming: mastering open source tools including Ant, JUnit, and Cactus /

Richard Hightower, Nicholas Lesiecki.
John Wiley & Sons,, New York : (2002)
0-471-20708-X

[5 Recomendado] Programador Java certificado: curso práctico /

Antonio J. Martín Sierra.
Ra-Ma,, Paracuellos de Jarama, Madrid : (2010) - (3ª ed.)
978-84-7897-972-1

[6 Recomendado] Test-driven development: a practical guide /

Dave Astels.
Prentice Hall PTR,, Upper Saddle River, N.J : (2003)
978-0-13-101649-1

[7 Recomendado] Ingeniería del software: un enfoque práctico /

Roger S. Pressman.

McGraw Hill,, México [etc.] : (2010) - (7ª ed.)

9786071503145