GUÍA DOCENTE 2018/19

Centro	310 - Facultad de Ciencia y Tecnología	**Ciclo**	Indiferente
Plan	GELECT30 - Grado en Ingeniería Electrónica	**Curso**	1er curso

ASIGNATURA

26662 - Fundamentos de Programación **Créditos ECTS :** 6

DESCRIPCIÓN Y CONTEXTUALIZACIÓN DE LA ASIGNATURA

Se parte de unos conocimientos y habilidades básicos de programación, como los adquiridos en la asignatura "Introducción a la computación". También se supone el dominio de conceptos matemáticos básicos sobre álgebra lineal, geometría y análisis. A partir de ahí, se presentan algoritmos básicos de búsqueda y ordenación, así como técnicas de análisis de la eficiencia. Se estudian tipos abstractos de datos en orden creciente de complejidad, así como algunas técnicas de diseño de algoritmos. Los ejemplos y ejercicios se apoyan en un lenguaje de programación de alto nivel utilizado en el entorno científico-tecnológico actual. La asignatura proporciona los conocimientos y habilidades necesarios para resolver problemas algorítmicos de complejidad media. Desde este punto de vista, la asignatura puede calificarse como auxiliar/instrumental, ya que proporciona las competencias necesarias para acometer tareas de modelado y simulación en muchas otras asignaturas del Grado.

COMPETENCIAS / RESULTADOS DE APRENDIZAJE DE LA ASIGNATURA

COMPETENCIAS ESPECÍFICAS
- Conocer las bases de la programación actual: organización de los datos, programación estructurada y programación orientada a objetos.
- Conocer y aplicar métodos de estimación de la complejidad computacional de un algoritmo.
- Ser capaz de aplicar una rigurosa metodología de programación basada en el conocimiento de las estructuras de datos y las primitivas de computación, así como el desarrollo de prácticas y trabajos relacionados.
- Conocer un lenguaje de programación actual y saber utilizarlo para la implementación de algoritmos básicos.
- Ser capaz de exponer de manera clara y concisa los programas realizados y las decisiones de diseño tomadas.

RESULTADOS DE APRENDIZAJE
- Conocer las características esenciales y las implementaciones más comunes de los tipos abstractos de datos básicos: lineales (pilas, colas y listas) y no lineales (tablas asociativas, árboles y grafos), para identificar en qué situaciones y de qué forma pueden utilizarse en diseños más generales.
- Conocer y aplicar las técnicas básicas de análisis de la complejidad computacional, para comparar varios algoritmos entre sí y elegir el más adecuado a un problema y un contexto determinados.
- Diseñar y reutilizar tipos abstractos de datos y aplicar técnicas básicas de diseño de algoritmos para resolver problemas de una manera estructuralmente clara y eficiente.
- Trabajar en equipo en un entorno real de programación en un lenguaje de programación de alto nivel para resolver un problema algorítmico, analizando las alternativas de solución, identificando los tipos abstractos de datos necesarios, reutilizando los que estén disponibles, diseñando e implementando el resto, y elaborando tablas de datos (perfiles de ejecución) que permitan tomar una decisión sobre cuál es la mejor solución en la práctica.

CONTENIDOS TEORICO-PRACTICOS

CONTENIDOS TEORICOS

Tema 1. Algoritmos de búsqueda y ordenación
Esquemas básicos de búsqueda: búsqueda secuencial y búsqueda binaria
Esquemas básicos de ordenación: inserción, selección e intercambio
Ordenación por partición
Ordenación por mezcla

Tema 2. Análisis de la eficiencia computacional de algoritmos
Notación asintótica frente a perfil de ejecución
Análisis de las estructuras de control
Análisis de algoritmos recursivos
Algoritmos de tipo Divide y Vencerás

Tema 3. Tipo Abstracto de Datos (TAD)
Diseño basado en TAD
Programación orientada a objetos: conceptos generales
Casos prácticos

Tema 4. TAD lineales
Pilas

Colas
Colas de Prioridad

Tema 5. TAD no lineales
Tablas asociativas
Montículos
Arboles
Arboles binarios de búsqueda

Tema 6. TAD Grafo (tema avanzado)
Definiciones, operaciones e implementaciones
Recorridos y conectividad
Arboles de recubrimiento de coste mínimo
Algoritmos voraces
Caminos de coste mínimo
Programación dinámica

CONTENIDOS PRACTICOS

Se proponen 3 enunciados abiertos (de complejidad creciente) para la resolución de problemas relacionados con los temas desarrollados en las clases teóricas. Los estudiantes deberán codificar una o varias soluciones y elaborar, dependiendo del caso, un breve informe (resultados, costes computacionales, etc.), que serán enviados a través de la plataforma e-gela. Los enunciados irán cambiando curso a curso, pero sus objetivos generales serán: (1) consolidación de los conocimientos adquiridos en "Introducción a la computación" en cuanto a estructuras de datos y lenguaje de programación; (2) estudio de la eficiencia computacional desde un punto de vista práctico (perfiles de ejecución); y (3) diseño, desarrollo y aplicación de uno o varios TAD en una situación realista.

METODOLOGÍA

La metodología docente hace uso de 5 vías de interacción entre el equipo docente y los/las estudiantes:

(1) Las clases magistrales, en las que el profesor o profesora expone un tema apoyándose en una presentación tipo PowerPoint, con apuntes en la pizarra y frecuentemente con el desarrollo de código en el propio ordenador. Aunque las clases permiten la interacción, y de hecho, se invita a los/las estudiantes a que pregunten cualquier duda que pueda surgir, se trata de una vía de aprendizaje fundamentalmente unidireccional. El material de apoyo a estas clases, el código desarrollado, los ejercicios propuestos, e incluso enlaces relevantes para la profundización en cada tema, se suministran a través de la plataforma e-gela.

(2) Las clases de problemas, en las que los/las estudiantes, con ayuda del profesor o profesora, presentan y comentan sus soluciones a los ejercicios, que han enviado a través de la plataforma e-gela. Eventualmente, estas clases también se utilizan para resolver dudas sobre los contenidos o sobre las prácticas de la asignatura. A través de la plataforma e-gela se suministran soluciones comentadas a los ejercicios trabajados. Esta modalidad docente es esencialmente interactiva.

(3) Las prácticas de programación, en las que los/las estudiantes, con el apoyo del equipo docente, codifican y depuran el código de la solución a un problema, en 7 sesiones presenciales que tienen lugar en un laboratorio de ordenadores. A lo largo del curso se proponen 3 enunciados de complejidad creciente (suministrados, junto a los conjuntos de datos necesarios, a través de la plataforma e-gela), que desarrollan aspectos concretos de las clases teóricas. Los/Las estudiantes deben codificar las soluciones y, dependiendo del caso, elaborar un breve informe. A través de la interacción permanente con el entorno de programación, con el equipo docente y con otros/otras estudiantes, estas sesiones prácticas tratan de prestar a los/las estudiantes desenvoltura y confianza en la aplicación de los conocimientos adquiridos y desarrollar habilidades creativas.

(4) Los seminarios, en los que se exponen aspectos de índole fundamentalmente práctica: temas avanzados del lenguaje de programación utilizado en la asignatura y del entorno de programación en el que se desarrollan las prácticas. Se trata de 5 sesiones de una hora de duración, que tienen lugar en un laboratorio de ordenadores, justo antes de las 5 primeras sesiones de prácticas, a las que sirven de apoyo/complemento.

(5) Las tutorías, en las que el profesor o profesora atiende a los/las estudiantes en su despacho, para resolver dudas sobre los contenidos, sobre los ejercicios propuestos o sobre las prácticas. Esta modalidad docente es la que permite una interacción más directa y personalizada. Existe una franja horaria oficial destinada a tutorías. Fuera de esa franja, los/las estudiantes son atendidos/as igualmente, previa cita, siempre que el profesor o profesora esté disponible. Por último, a lo largo del curso los/las estudiantes pueden plantear dudas sobre cualquier aspecto de la asignatura también a través del e-mail, dudas que generalmente son atendidas en un plazo no superior a dos días.

TIPOS DE DOCENCIA

Tipo de Docencia	M	S	GA	GL	GO	GCL	TA	TI	GCA
Horas de Docencia Presencial	30	5	10		15				
Horas de Actividad No Presencial del Alumno	45	7,5	15		22,5				

Leyenda: M: Magistral S: Seminario GA: P. de Aula GL: P. Laboratorio GO: P. Ordenador GCL: P. Clínicas TA: Taller TI: Taller Ind. GCA: P. de Campo

SISTEMAS DE EVALUACIÓN

- Sistema de evaluación continua
- Sistema de evaluación final

HERRAMIENTAS Y PORCENTAJES DE CALIFICACIÓN

- Prueba escrita a desarrollar 60%
- Trabajos individuales 20%
- Trabajos en equipo (resolución de problemas, diseño de proyectos) 20%

CONVOCATORIA ORDINARIA: ORIENTACIONES Y RENUNCIA

En la vía de EVALUACIÓN CONTINUA (que será la vía de evaluación por defecto en esta asignatura), el cómputo de la nota final será como sigue:
- Examen escrito: 60%
- Prácticas (entrega de informes en plazo, y en su caso, explicación o defensa): 20%
- Trabajos individuales (entrega de ejercicios resueltos en plazo, y en su caso, explicación o defensa): 20%

Aquellos/as estudiantes que deseen seguir la vía de EVALUACIÓN FINAL deberán presentar al profesor/a por escrito su renuncia a la evaluación continua, como muy tarde en la semana 9 del cuatrimestre.

El cómputo de la nota final en la vía de evaluación final será como sigue:
- Examen escrito: 60%
- Examen de laboratorio: 40%

La hora, el lugar de celebración y las demás condiciones del examen de laboratorio se comunicará a los/las estudiantes con al menos UN MES de antelación.

En todo caso, para aprobar la asignatura será necesario obtener al menos un 4 sobre 10 en el examen escrito.

CONVOCATORIA EXTRAORDINARIA: ORIENTACIONES Y RENUNCIA

En convocatoria extraordinaria, tal como establece la Normativa UPV/EHU, se aplicará exclusivamente el sistema de EVALUACIÓN FINAL, en dos posibles modalidades:

* MODALIDAD A: se mantienen las calificaciones de prácticas y trabajos individuales obtenidas durante el curso, de modo que el/la estudiante simplemente se presenta al examen escrito. El cómputo de la nota final será como sigue:
- Examen escrito: 60%
- Prácticas (entrega de informes en plazo, y en su caso, explicación o defensa): 20%
- Trabajos individuales (entrega de ejercicios resueltos en plazo, y en su caso, explicación o defensa): 20%

* MODALIDAD B: es la misma vía de evaluación final definida para la convocatoria ordinaria. El cómputo de la nota final será como sigue:
- Examen escrito: 60%
- Examen de laboratorio: 40%

La hora, el lugar de celebración y las demás condiciones del examen de laboratorio se comunicará a los/las estudiantes con al menos 10 DÍAS de antelación.

La elección de la MODALIDAD B deberá comunicarse por escrito al profesor/a al menos 14 DIAS ANTES de la fecha establecida para el examen escrito. De no recibir ninguna comunicación, se entenderá que el/la estudiante elige la MODALIDAD A.

En todo caso, para aprobar la asignatura será necesario obtener al menos un 4 sobre 10 en el examen escrito.

MATERIALES DE USO OBLIGATORIO

BIBLIOGRAFIA

Bibliografía básica

1. Gilles Brassard, Paul Bratley. Fundamentos de algoritmia. Prentice-Hall, 1997.
2. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. Introduction to Algorithms (Third Edition). The MIT Press, 2009.
3. Bradley N. Miller, David L. Ranum. Problem Solving with Algorithms and Data Structures Using Python (Second Edition). Franklin, Beedle & Associates, 2011.
4. Rance D. Necaise. Data Structures and Algorithms Using Python. John Wiley & Sons, 2011.
5. Mark Summerfield. Programming in Python 3. A Complete Introduction to the Python Language (Second Edition). Addison-Wesley Professional, 2010.

Bibliografía de profundización

6. Narciso Martí, Yolanda Ortega, José Alberto Verdejo. Estructuras de datos y métodos algorítmicos: ejercicios resueltos. Prentice Hall, 2004.
7. Steven S. Skiena. The Algorithm Design Manual (Second Edition). Springer, 2008.
8. Vernon L. Ceder. The Quick Python Book (Second Edition). Manning Publications, 2010.
9. David M. Beazley. Python Essential Reference (4th Edition). Addison-Wesley Professional, 2009.
10. Mark Lutz. Learning Python (Fifth Edition). O'Reilly Media, 2013.

Revistas

Direcciones de internet de interés

Problem Solving with Algorithms and Data Structures Using Python - Official Website
http://interactivepython.org/runestone/static/pythonds/index.html

Python Programming Language - Official Website
http://python.org/

Python 3 documentation
https://docs.python.org/3/

The Python 3 Tutorial
https://docs.python.org/3/tutorial/

OBSERVACIONES