

Ingeniería del Rendimiento

Código: 104341
Créditos ECTS: 6

Titulación	Tipo	Curso	Semestre
2503758 Ingeniería de Datos	OB	2	1

La metodología docente y la evaluación propuestas en la guía pueden experimentar alguna modificación en función de las restricciones a la presencialidad que impongan las autoridades sanitarias.

Contacto

Nombre: Juan Carlos Moure Lopez
Correo electrónico: JuanCarlos.Moure@uab.cat

Uso de idiomas

Lengua vehicular mayoritaria: español (spa)
Algún grupo íntegramente en inglés: No
Algún grupo íntegramente en catalán: No
Algún grupo íntegramente en español: No

Equipo docente

Juan Carlos Moure Lopez

Prerequisitos

Es muy recomendable haber cursado y aprobado las asignaturas de Fundamentos de Informática, Fundamentos de Programación y Programación Avanzada

Objetivos y contextualización

1. Evaluar y analizar el rendimiento de la ejecución de un programa.
2. Entender la conversión desde el lenguaje fuente (Python / C) al ensamblador, identificando las transformaciones en el código.
3. Determinar las dependencias de datos en los lazos de ejecución de los programas y usar estrategias para aumentar el paralelismo de instrucciones y de datos.
4. Comprender el funcionamiento de la jerarquía de memoria y utilizar estrategias para aumentar la localidad de acceso a los datos.

Competencias

- Buscar, seleccionar y gestionar de manera responsable la información y el conocimiento.
- Diseñar soluciones algorítmicas eficientes para problemas computacionales, implementarlas en forma de desarrollo de software robustos, estructurados y fáciles de mantener, y verificar su validez.
- Evaluar de manera crítica el trabajo realizado.
- Planificar y gestionar el tiempo y los recursos disponibles.
- Que los estudiantes hayan demostrado poseer y comprender conocimientos en un área de estudio 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.

Resultados de aprendizaje

1. Analizar el rendimiento de aplicaciones, detectar cuellos de botella y aplicar posibles optimizaciones.
2. Aplicar los principios básicos de la estructura y la programación de los computadores.
3. Buscar, seleccionar y gestionar de manera responsable la información y el conocimiento.
4. Evaluar de manera crítica el trabajo realizado.
5. Implementar en un lenguaje de programación problemas algorítmicos de dificultad media.
6. Planificar y gestionar el tiempo y los recursos disponibles.
7. Que los estudiantes hayan demostrado poseer y comprender conocimientos en un área de estudio 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.

Contenido

1. Evaluación del Rendimiento: herramientas para monitorizar e instrumentar la ejecución de un programa.
2. Análisis del Rendimiento: complejidad computacional, factores determinados por la arquitectura del procesador y métricas de rendimiento.
3. Paralelismo en la ejecución de Instrucciones: Pipeline y Vectorización.
4. Análisis de Dependencias de datos en los lazos de ejecución.
5. Jerarquía de Memoria y Localidad de acceso a los datos: efecto en el rendimiento.
6. Estrategias algorítmicas de optimización: aumentar el paralelismo y la localidad de acceso a los datos.

Metodología

Clases de Teoría: se expondrán los conocimientos propios de la asignatura. Se describirán los conceptos básicos y se ilustrarán con ejemplos prácticos. Se resaltarán los problemas de aprendizaje más importantes y se mostrará cómo completar y profundizar en los contenidos. Se discutirán casos prácticos y el profesor detectará los problemas de comprensión y razonamiento más comunes y los resolverá para todos los estudiantes.

Clases de Problemas en Laboratorio: se harán actividades cooperativas de resolución de problemas y de casos prácticos sencillos, que servirán de apoyo a la teoría. Tras un trabajo individual previo, se hará una puesta en común en grupo y se resolverán las dudas. El profesor detectará los problemas de comprensión y razonamiento más comunes y los resolverá por grupos o para todo el alumnado. Se promueve la capacidad de análisis y síntesis del alumno, el razonamiento crítico y la capacidad de resolución de problemas. Durante la sesión los alumnos y alumnas deberán informar al profesor sobre sus avances y los problemas que se puedan encontrar, y entregarán un documento final con los resultados y reflexiones sobre sus actividades.

Problemas de Optimización: durante todo el curso se propondrán ejercicios para transformar pequeños fragmentos de código de forma que se ejecuten en menos tiempo. Los alumnos y alumnas deben aplicar los conocimientos, metodologías y estrategias aprendidas y presentar el resultado argumentando las razones de las mejoras en el rendimiento. Se espera por parte de los estudiantes una actitud proactiva y dinámica, y la búsqueda autónoma de los conocimientos que sean necesarios.

Nota: se reservarán 15 minutos de una clase dentro del calendario establecido por el centro o por la titulación para que el alumnado rellene las encuestas de evaluación de la actuación del profesorado y de evaluación de la asignatura o módulo.

Actividades

Título	Horas	ECTS	Resultados de aprendizaje
Tipo: Dirigidas			
Clases de Teoría	26	1,04	3, 7

Resolución de Problemas prácticos en el Laboratorio	26	1,04	1, 2, 4, 5, 6
Tipo: Supervisadas			
Optimización de Rendimiento de fragmentos de código	26	1,04	1, 2, 4, 3, 5, 6
Tipo: Autónomas			
Estudio autónomo	30	1,2	3, 6
Preparación previa y documentación del trabajo en el laboratorio	40	1,6	1, 4, 3, 6, 7

Evaluación

Las fechas de evaluación continua y entrega de trabajos se publicarán en la plataforma Moodle de la UAB (Campus Virtual) y pueden estar sujetas a cambios de programación por motivos de adaptación a posibles incidencias. Siempre se informará en la plataforma Moodle sobre estos cambios ya que se entiende que es la plataforma habitual de intercambio de información entre profesor y estudiantes.

Actividades de evaluación

Actividad A: resolución de problemas prácticos en el laboratorio; se realiza en grupo; supone un 50% de la nota final; es necesario obtener una nota mínima de 5 puntos sobre 10; y no se puede recuperar.

Actividad B: entrega de un informe de la actividad A; se realiza individualmente; supone un 15% de la nota final; no es necesario obtener una nota mínima; y no se puede recuperar.

Actividad C: asistencia y participación activa en clase; se realiza individualmente; supone un 5% de la nota final; no es necesario obtener una nota mínima; y no se puede recuperar.

Actividad D: examen escrito de teoría y problemas; se realiza individualmente y sin apuntes; supone un 30% de la nota final; es necesario obtener una nota mínima de 5 puntos sobre 10; y se puede recuperar.

Aparte de las actividades de evaluación descritas anteriormente, durante el curso será posible proponer otras actividades voluntarias que puedan contribuir a la calificación final del curso

Programa de Actividades de Evaluación

Las actividades A, B y C se realizan durante todo el curso. La actividad D se realiza al final del curso, en las fechas especificadas por la coordinación.

Proceso de Recuperación

El y la estudiante se puede presentar a la recuperación siempre que se haya presentado a las actividades A, C y D. Solamente se podrá recuperar la actividad D, que representa el 30% de la nota final.

Procedimiento de Revisión

Todas las actividades de evaluación se podrán revisar en un lugar, fecha y hora que se publicará con anterioridad. Se podrán hacer reclamaciones sobre la nota de la actividad, que serán evaluadas por el profesorado responsable de la asignatura. Si la o el estudiante no se presenta a esta revisión, no se revisará posteriormente esta actividad.

Calificaciones

Se otorgará una calificación de matrícula de honor (MH) a los estudiantes que alcancen una nota final igual o superior a 9,00, hasta un total del 5% de los estudiantes matriculados, priorizando la parte de la nota que corresponde a evaluaciones individuales (actividades B, C y D).

Si el alumno o alumna no alcanza la nota mínima en alguna de las actividades de evaluación, la calificación numérica final será el mínimo entre 4,5 puntos y la suma ponderada de las notas de todas las actividades.

Un o una estudiante se considerará no evaluable (NA) si no se ha presentado a ninguna de las actividades A, B y D.

Irregularidades por parte de los estudiantes, copia y plagio

Sin perjuicio de otras medidas disciplinarias que se estimen oportunas, se calificarán con un cero la irregularidades cometidas por un estudiante que puedan conducir a una variación de la calificación de un acto de evaluación. Por tanto, la copia, el plagio, el engaño, dejar copiar, etc. en cualquiera de las actividades de

evaluación implicará suspenderla con un cero. Las actividades de evaluación calificadas de esta forma y por este procedimiento no serán recuperables. Si es necesario superar cualquiera de estas actividades de evaluación para aprobar la asignatura, entonces esta asignatura quedará suspendida directamente, sin oportunidad de recuperarla en el mismo curso. En el caso de no superar la asignatura debido a que el estudiante ha cometido irregularidades en un acto de evaluación, la nota numérica del expediente será el valor menor entre 3.0 y la media ponderada de las notas (y por tanto no será posible el aprobado por compensación).

Evaluación de los estudiantes repetidores

No hay ningún tratamiento especial para los estudiantes repetidores.

Actividades de evaluación

Título	Peso	Horas	ECTS	Resultados de aprendizaje
Examen escrito individual de teoría y problemas	30%	2	0,08	1, 4, 7
Informes individuales de la resolución de problemas prácticos en el laboratorio	15%	0	0	1, 4, 3, 7
Participación en clase/laboratorio	5%	0	0	1, 4, 3, 6, 7
Trabajo en grupo para resolver problemas prácticos en el laboratorio + Respuestas + Informes	50%	0	0	1, 2, 3, 5, 6, 7

Bibliografía

Computer Architecture: A Quantitative Approach. 6th Edition

(https://cataleg.uab.cat/iii/encore/record/C__Rb2034006)

John Hennessy and David Patterson, Morgan Kaufmann (Elsevier) 2018 (Cap. 1-3)

Computer Systems: A Programmer's Perspective. 3rd Edition (North-American)

(https://csuc-uab.primo.exlibrisgroup.com/permalink/34CSUC_UAB/avjcib/alma991006392489706709)

Randall E. Bryant and David R. O'Hallaron, Addison Wesley, 2015 (Cap. 5 i 6)

Computer Organization and Design RISC-V Edition: The Hardware/Software Interface.

David Patterson and John Hennessy, Morgan Kauffman (Elsevier), 2017 (Cap. 4 i 5)

High Performance Python: practical performant programming for human.

Micha Gorelick and Ian Ozsvald, O'Reilly, 2014

Cython: a guide for Python programmers.

Kurt W. Smith, O'Reilly, 2015

Python High Performance programming. Boost the performance of your Python programs using advanced techniques

Gabriele Lanaro, Packt Publishing, 2013

Software

Ninguno