Grado en Matemáticas
Programación Paralela

CRÉDITOS PRESENCIALES: 6

CRÉDITOS NO PRESENCIALES:

SEMESTRE: 2

BREVE DESCRIPTOR El curso proporciona una introducción a la programación en entornos en los que hay múltiples procesadores interconectados entre sí. Se hace un recorrido general y en anchura por las diversas técnicas de programación que se emplean en resolver problemas prototípicos que surgen en este tipo de sistemas: sincronización de tareas, paralelización de soluciones, problemas de consenso...

También se adentra en las cuestiones técnicas necesarias de un lenguaje de programación concreto, *Python*, para poder programar los algoritmos y diseños estudiados.

REQUISITOS Es adecuado para seguir el curso con fluidez contar con los conocimientos de programación que habitualmente se consiguen con un primer curso en esta materia: variables y tipos de datos, estructuras de control, programación estructurada...

OBJETIVOS Adquirir los conocimientos elementales y la terminología adecuada que rodea al mundo de la programación paralela/concurrente/distribuida.

Entender los múltiples factores que intervienen en la definición de un problema –comunicación, topología, dependencias...– y cuyo estudio es fundamental para afrontar una solución.

Entender la importancia del correcto diseño –complejidad algorítmica, adecuación, escalabilidad...– y el estudio teórico en la resolución de problemas mediante programas.

Utilización de los módulos existentes en Python para diseñar e implementar soluciones a problemas concurrentes, paralelos y distribuidos.

Comprender el uso potencial de la programación paralela en multitud de ambitos reales: simulación, la computación científica, gráficos y visión por ordenador, entornos colaborativos, web...

COMPETENCIAS

Generales Utilizar los conocimientos matemáticos para modelizar y resolver problemas complejos mediante algoritmos.

Valorar diferentes soluciones alternativas y elegir de acuerdo a las situaciones específicas entre las herramientas y las técnicas adecuadas para llevar a cabo la implementación de dichas soluciones.

Saber abstraer en un problema complejo las propiedades y características esenciales reconociendo su rango de aplicabilidad y limitaciones.

Transversales Ser capaz de mostrar creatividad, iniciativa y espíritu emprendedor para afrontar los retos de su actividad y saber valorar las soluciones a dichos retos en el contexto industrial, económico, administrativo, medio ambiental y social.

Tener la capacidad de reunir e interpretar datos relevantes para emitir juicios que incluyan una reflexión profunda sobre temas y problemas a resolver.

Demostrar razonamiento crítico y gestionar información científica y técnica de calidad, bibliografía, bases de datos especializadas y recursos accesibles a través de Internet.

CONTENIDOS TEMÁTICOS

- Conceptos y Teminología
- Distintos Modelos y Arquitecturas Alternativas
- Sincronización de Tareas
- Algoritmos Distribuidos
- Programación Multiprocesador
- Clusters y Grids
- Aplicaciones Prácticas

ACTIVIDADES DOCENTES

Clases teóricas: En las que el profesor presenta los conceptos y técnicas relevantes de los contenidos y muestra las referencias bibliográficas o enlaces web a seguir para profundizar en dichos temas.

Clases prácticas: En las que en un trabajo guiado por el profesor el alumnado en grupo diseña y busca soluciones a ejercicios propuestos.

Laboratorios: En las que los alumnos de forma individual implementan y depuran los programas que resuelven los problemas estudiados en las clases prácticas.

Presentaciones: En las que los alumnos, de forma individual o por grupos, a petición del profesor, preparan una exposición ante sus compañeros de una solución –bien sea diseño, implementación o ambas– a alguno de los problemas propuestos.

EVALUACIÓN La calificación final proviene de la evaluación continuada basada en la participación en clases y laboratorios, así como en la entrega de prácticas de programación. Para los alumnos que no hayan superado la evaluación continuada podrán hacer un examen final y una práctica final que, ponderados al 50 % darán la calificación definitiva.

BIBLIOGRAFÍA BÁSICA

- M. Ben-Ari, *Principles of Concurrent and Distributed Programming*, Prentice-Hall, 2006.
- Thomas Rauber and Gudula Rünger, *Parallel Programming For Multicore and Cluster Systems*, Springer-Verlag, 2010.
- Timothy G. Mattson, Beverly A. Sanders and Berna L. Massingill, *Patterns for Parallel Programming*, Addison-Wesley, 2005.
- Nancy A. Lynch, *Distributed Algorithms*, Morgan Kaufmann, 1996.
- Documentación de Python, `http://docs.python.org/release/2.7.1/`