

PROGRAMA DE CURSO

Código	Nombre			
CC4102	Diseño y Análisis de Algoritmos			
Nombre en Inglés				
Design and Analysis of Algorithms				
SCT	Unidades Docentes	Horas de Cátedra	Horas Docencia Auxiliar	Horas de Trabajo Personal
6	10	3	1.5	5.5
Requisitos			Carácter del Curso	
FI2003, (CC3102/ MA5201/ Autor), (MA3403/MA3401)			Obligatorio para la ICC	
Resultados de Aprendizaje				
<p>El curso entrega herramientas avanzadas para el diseño y análisis de algoritmos, incluyendo mecanismos para demostrar cotas inferiores, para diseño en memoria secundaria, para análisis amortizado, y para diseño y análisis de algoritmos probabilísticos, aproximados y paralelos.</p> <p>El alumno que apruebe el curso habrá demostrado:</p> <ul style="list-style-type: none"> - Manejar nociones básicas de diseño, interpretación y presentación de resultados experimentales. - Comprender el concepto de complejidad de un problema como cota inferior, y conocer técnicas elementales para demostrar cotas inferiores. - Comprender las técnicas de diseño de algoritmos y estructuras de datos para memoria secundaria. - Comprender el concepto de análisis amortizado de algoritmos. Ser capaz de diseñar algoritmos y estructuras de datos considerando costo amortizado, y ser capaz de analizar este costo. - Ser capaz de diseñar algoritmos eficientes usando la finitud del dominio. Poder distinguir estas situaciones. - Comprender el concepto de algoritmos en línea y competitividad. Ser capaz de diseñar algoritmos en línea y analizar su competitividad. - Comprender el concepto de algoritmos aleatorizados y probabilísticos, y cuándo son relevantes. Ser capaz de diseñar y analizar algoritmos de este tipo. - Comprender el concepto de algoritmos aproximados, y cuándo son relevantes. Ser capaz de diseñar y analizar algoritmos de este tipo. - Comprender el concepto de paralelización en algoritmos. Ser capaz de diseñar y analizar algoritmos paralelos. - Conocer un conjunto significativo de algoritmos y estructuras de datos de mediana complejidad para solución de problemas básicos. 				

Metodología Docente	Evaluación General
<p>Clases expositivas del profesor de cátedra, buscando la participación de los alumnos en pequeños problemas que se van proponiendo durante la exposición.</p> <p>Clases auxiliares dedicadas a explicar ejemplos más extensos, resolver ejercicios propuestos, y preparación pre y post controles.</p> <p>Cuando el tiempo lo permite, exposición de las mejores tareas de los alumnos, como casos de estudio de implementación y experimentación.</p>	<p>Se realizan tres controles para evaluar si se han cumplido los objetivos. El primero evalúa las unidades 1 y 2, el segundo la unidad 3 y el tercero la unidad 4. El examen evalúa todas las unidades, en particular la parte de la 4 que no llega a evaluarse completamente en el control. Tanto los controles como el examen se enfocan con igual peso a evaluar que el alumno haya comprendido los conceptos como los casos de estudio más significativos utilizados para ilustrarlos, pues éstos tienen valor intrínseco en la formación.</p> <p>La Nota de Control se calcula de la siguiente manera: Promedio ponderado del examen (40%) y del promedio de los controles (60%).</p> <p>Se realizan asimismo tres tareas a lo largo del curso. Las tareas consisten generalmente en implementar algoritmos o estructuras de datos alternativos para resolver un cierto problema, comparando las soluciones básicas con las que se introducen en el curso. Las tareas buscan fundamentalmente que el alumno comprenda la diferencia entre teoría y práctica, que implemente soluciones de mediana complejidad vistas sólo en la pizarra, y que se enfrente al problema de diseñar e interpretar experimentos. Las tareas se realizan en grupos de 1 o 2 personas y se promedian a partes iguales para formar la nota de tareas. Se elegirán las mejores tareas para que sus autores las vayan exponiendo a lo largo del semestre, de forma de entregar experiencias directas de casos de estudio a sus compañeros y recibir preguntas y retroalimentación de éstos.</p> <p>Controles y tareas se aprueban por separado y deben ser iguales o superiores a 4.0. La nota final es $\frac{2}{3}$ de la nota de controles y $\frac{1}{3}$ de la nota de tareas.</p>

Unidades Temáticas

Número	Nombre de la Unidad	Duración en Semanas	
1	Conceptos básicos y complejidad	3	
Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía	
<p>1. Repaso del proceso de diseño y análisis de un algoritmo. Metodología de experimentación.</p> <p>2. Técnicas para demostrar cotas inferiores: adversario, teoría de la información, reducción.</p> <p>3. Principales casos de estudio (o similares): cota inferior para mínimo y máximo de un arreglo, cota inferior para búsqueda en un arreglo con distintas probabilidades de acceso, cota inferior para la cápsula convexa, 3SUM reducido a puntos colineales, árboles de búsqueda óptimos.</p>	<p>Adquirir nociones básicas de experimentación en algoritmos. Comprender el concepto de complejidad de un problema como cota inferior, y conocer técnicas elementales para demostrar cotas inferiores. Conocer algunos casos de estudio relevantes.</p>	<p>[1] Cap 1-4. [3] Cap 2, 10. [4] Cap 1, 4. [5] Cap 2. [6] Cap 1-2, 10. [7] Cap 6. [9] Cap 8-9. [10] Cap 1,2.</p>	

Número	Nombre de la Unidad	Duración en Semanas	
2	Algoritmos y Estructuras de Datos para Memoria Secundaria	3	
Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía	
<p>1. Modelo de computación en memoria secundaria. Accesos secuenciales y aleatorios</p> <p>2. Ordenamiento en memoria secundaria: Mergesort. Cota inferior.</p> <p>3. Colas de prioridad en memoria secundaria. Cotas inferiores.</p> <p>4. Diccionarios en memoria secundaria: repaso de árboles B, hashing lineal y extensible.</p>	<p>Comprender el modelo de costo de memoria secundaria. Conocer algoritmos y estructuras de datos básicos que son eficientes en memoria secundaria, y el análisis de su desempeño.</p>	<p>[1] Cap 18. [5] Cap 4.7, 7.11. [7] Cap 13, 18. [9] Cap 11. [10] Cap 5.7, 6.3.</p>	

Número	Nombre de la Unidad	Duración en Semanas
3	Técnicas avanzadas de diseño y análisis de algoritmos	4
Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
<ol style="list-style-type: none"> 1. Análisis amortizado de algoritmos y estructuras de datos: análisis completo, contabilidad de costos, función potencial. 2. Uso de dominios discretos y finitos en el diseño de algoritmos. 3. Algoritmos en línea. Competitividad. 4. Principales casos de estudio (o similares): estructuras para union-find, colas binomiales, splay trees, radix sort, árboles de van Emde Boas, árboles Patricia y de sufijos, paginamiento, move-to-front, búsqueda no acotada, k-server problem. 	<p>Comprender las técnicas de algoritmos de costo amortizado, uso de finitud, y algoritmos competitivos. Ser capaz de diseñar y analizar algoritmos y estructuras de datos basados en estos principios. Conocer algunos casos de estudio relevantes.</p>	<p>[1] Cap 8, 17, 19, 21. [2] Cap 4. [3] Cap 6. [5] Cap 4-6, 8, 11. [7] Cap 17. [8] Cap 9. [9] Cap 5. [10] Cap 3.3.</p>

Número	Nombre de la Unidad	Duración en Semanas
4	Algoritmos no convencionales	5
Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
<ol style="list-style-type: none"> 1. Algoritmos aleatorizados y probabilísticos. Casos en que se justifican. Relación con la NP-completitud. 2. Algoritmos tipo Monte Carlo y Las Vegas. 3. Aleatorización de la entrada. Independencia de la distribución de la entrada. Estructuras de datos aleatorizadas. 4. Solución de problemas NP-completos: búsqueda exhaustiva. Concepto de algoritmos aproximados. 5. Nociones de aproximabilidad. Problemas que son o no aproximables. 	<p>Comprender el concepto de algoritmos aleatorizados, probabilísticos, aproximados, y paralelos, y cuándo son relevantes. Ser capaz de diseñar y analizar algoritmos de estos tipos. Conocer algunos casos de estudio relevantes.</p>	<p>[1] Cap 5, 31.8, 32.2, 35. [3] Cap 6, 11, 12. [4] Cap 6. [5] Cap 10. [6] Cap 8. [7] Cap 35, 40, 44. [8] Cap 1, 7, 8, 12, 14. [10] Cap 4, 12.</p>

<p>6. Algoritmos paralelos y distribuidos. Medidas de complejidad. Técnicas de diseño.</p> <p>7. Principales casos de estudio (o similares): primalidad, árboles binarios de búsqueda aleatorizados, quicksort, hashing universal y perfecto, aproximaciones para recubrimiento de vértices, vendedor viajero, mochila. Ordenamiento paralelo, parallel-prefix, list ranking.</p>		
---	--	--

Bibliografía

- [1] T. Cormen, C. Leiserson, R. Rivest, C. Stein. Introduction to Algorithms, 3rd edition. MIT Press, 2009.
- [2] A. Aho, J. Hopcroft, J. Ullman. The Design and Analysis of Computer Algorithms. Addison-Wesley, 1974.
- [3] U. Manber. Introduction to Algorithms. Addison-Wesley, 1989.
- [4] G. Rawlins. Compared to what? Computer Science Press, 1992.
- [5] M. Weiss. Data Structures and Algorithm Analysis, 2nd edition. Benjamin Cummings, 1995.
- [6] G. Brassard, P. Bratley. Algorithmics. Theory and Practice. Prentice-Hall, 1988.
- [7] R. Sedgewick. Algorithms in C++. Addison-Wesley, 1992.
- [8] R. Motwani, P. Raghavan. Randomized Algorithms. Cambridge, 2000.
- [9] A. Aho, J. Hopcroft, J. Ullman. Data Structures and Algorithms. Addison-Wesley, 1983.
- [10] K. Mehlhorn and P. Sanders. Algorithms and Data Structures. Springer, 2008.
- [11] S. Skiena. The Algorithm Design Manual. Springer, 2008.
- [12] M. Mitzenmacher y E. Upfal. Probability and Computing. Cambridge, 2005.
- [13] N. Ziviani. Diseño de Algoritmos. Thomson, 2007.
- [14] A. Levitin. The Design & Analysis of Algorithms, 2nd ed. Pearson, 2007.
- [15] R. Lee, S. Tseng, R. Chang, T. Tsai. Introduction to the Design and Analysis of Algorithms. McGraw-Hill, 2005.
- [16] J. Kleinberg y E. Tardos. Algorithm Design. Addison-Wesley, 2006.
- [17] J. Jája. An Introduction to Parallel Algorithms. Addison-Wesley, 1992.
- [18] A. Borodin y R. El-Yaniv. Online Computation and Competitive Analysis. Cambridge, 2005.

Vigencia desde:	Primavera 2016
Elaborado por:	Gonzalo Navarro