

UNIVERSIDAD NACIONAL DE RIO CUARTO
FACULTAD DE CIENCIAS EXACTAS, FÍSICO-QUÍMICAS Y NATURALES
DEPARTAMENTO DE COMPUTACION

Carreras: Analista en Computación y Profesorado en Ciencias de la Computación (Estructuras de Datos y Algoritmos), Licenciatura en Ciencias de la Computación (Algoritmos I)

Plan de Estudio: 1999

Asignaturas: Estructura de Datos y Algoritmos/Algoritmos I **Códigos:** 3325/3301

Docente Responsable: Dr Pablo Castro

Equipo Docente: Lic. Sonia Permigiani, Prof Ernesto Cerdá, Dr. Pablo Ponzio

Año Académico: 2016

Régimen de la Asignatura: Cuatrimestral

Régimen de Correlatividades:

Aprobada	Regular
Introduccion a la algoritmica y Programación	Programación Avanzada
	Algebra (Algoritmos I)

Carga Horaria Total: 112

Teóricas: 4 horas Semanales **Prácticas:** 4 Horas semanales **Laboratorios:** 4 Horas Semanales

Carácter de la Asignatura: Obligatoria

Contextualización de la Signatura: La asignatura está en el el segundo año de las carreras citadas, el material abordado en la materia provee las herramientas básicas para poder abordar los cursos más avanzados de computación.

Objetivos Propuestos:

El objetivo esencial de la materia es lograr que los alumnos se familiaricen con las técnicas más importantes de representación de datos, que comprendan los mecanismos de los lenguajes de programación modernos para su implementación, y sepan identificar los problemas prácticos en los cuales pueden emplearse cada una de las representaciones de datos estudiadas.

Un objetivo importante de la asignatura es que los alumnos se familiaricen con los conceptos fundamentales de la programación orientada a objetos, y realicen sus primeras actividades prácticas en

relación al diseño y la implementación de programas orientados a objetos.

Contenidos Básicos del Programa a Desarrollar:

Introducción a la programación orientada a objetos. La noción de tipos abstractos de datos. Revisión de implementaciones clásicas de tipos abstractos de datos estructurados lineales. Listas, pilas y colas, y sus implementaciones clásicas. Medidas de tiempo de ejecución de programas. Análisis de tiempo de ejecución en caso promedio y peor caso. Análisis de tiempo de ejecución de algoritmos iterativos y recursivos. Tipos abstractos de datos más avanzados. El tipo abstracto de datos Grafo. Algoritmos de recorrido/visita de grafos: recorridos "primero en profundidad" y "primero a lo ancho". Implementaciones clásicas de grafos (matrices de adyacencias y listas de adyacencias). Grafos dirigidos y no dirigidos. Grafos con costos. Aplicaciones del tipo abstracto de datos Grafo. Algoritmos clásicos sobre grafos. Cálculo de camino de mínimo costo: Algoritmos de Dijkstra y Floyd. Clausura transitiva: Algoritmo de Warshall. El problema del árbol abarcador de costo mínimo: Algoritmos de Prim y Kruskal. Estructuras de datos avanzadas. Heaps y sus aplicaciones.

Fundamentación de los Contenidos:

Los contenidos abordados en esta materia son los clásicos abordados en la bibliografía de estructuras de datos y algoritmos, estos proveen a los alumnos las herramientas básicas que les permitirá construir software de calidad en su vida profesional; así como se enseñarán los conceptos fundamentales para que los alumnos puedan comprender la teoría más avanzada en ciencias de la computación que será necesaria en su vida académica. Las estructuras de datos que son enseñadas proveen una visión amplia de las prácticas de programación existentes en la actualidad y además la teoría de complejidad abordada en la materia permite que los estudiantes incorporen criterios de calidad de software que son fundamentales en la actualidad. Los conceptos abordados son estudiados con un lenguaje orientado a objetos que es muy utilizado en la actualidad, el lenguaje Java.

Actividades a Desarrollar:

Clases teóricas: 4 horas semanales de clases teóricas, divididas en dos clases de dos horas cada una; en estas clases se les enseña a los estudiantes la teoría básica de la materia.

Clases práctica: 4 horas semanales de clases prácticas y de laboratorio, estas clases permiten a los estudiantes afianzar los conocimientos teóricos mediante la aplicación de la teoría a problemas medianamente complejos.

Laboratorios: 4 horas semanales de laboratorios, en estas clases los estudiantes implementan en lenguajes de programación los conceptos vistos en los teóricos y prácticos.

Horarios de Consulta: Cada docente provee horas de consulta adicionales para los prácticos, teóricos y laboratorios, estas son dos horas semanales adicionales.

Nómina de Trabajos Practicos. Se realizarán 10 trabajos prácticos para las clases de laboratorios y horas prácticas: Reuso, Encapsulación, Orientación a Objetos, Algebras, Intro.Java, Arboles, Análisis de Tiempo de Ejecución, Sorting, Diccionarios, Grafos,

Además se dará a los alumnos un trabajo práctico obligatorio sobre programación, que deberán aprobar para regularizar la materia.

Horarios de Clases:

Teóricos: Miércoles y Viernes de 14 a 16hs.

Prácticos: Miércoles de 10 a 12hs, Jueves de 14 a 16hs.

Laboratorios: Lunes de 10 a 12hs y Lunes de 14 a 16hs

Modalidad de Evaluación:

Exámenes parciales.

La materia tendrá dos exámenes parciales, cada uno con una recuperación. Los exámenes parciales serán teórico-prácticos, y abarcarán la primera y la segunda mitad de los contenidos de la asignatura, respectivamente. La evaluación del alumnado será complementada con trabajos prácticos obligatorios (ver más abajo).

Examen Final.

El examen final para alumnos regulares se llevará a cabo mediante evaluación oral, si el número de alumnos evaluados así lo permite. Abarcará la totalidad de los contenidos de la asignatura, verificando que los alumnos hayan adquirido los conocimientos teóricos y puedan aplicarlos en casos concretos en la práctica.

Por otra parte, el examen final para alumnos libres estará compuesto por una primera instancia escrita, en la cual se evaluarán las capacidades de los alumnos para construir programas utilizando las técnicas de representación de datos aprendidas en la asignatura. La segunda instancia del examen final para alumnos libres será similar al examen final para alumnos regulares.

Programa Analítico**Contenidos.**

Introducción a la programación orientada a objetos. Clases y objetos. Relaciones de agregación y uso. Herencia y polimorfismo. Genericidad. Introducción al diseño orientado a objetos. Conceptos de cohesión y acoplamiento. Principio abierto-cerrado.

La noción de tipos abstractos de datos. Representación de datos. Funciones de abstracción e invariantes de representación. Implementación de tipos abstractos de datos en lenguajes orientados a objetos. Especificaciones básicas y jerárquica. Corrección y completitud

Revisión de implementaciones clásicas de tipos abstractos de datos estructurados lineales. Listas, pilas y colas, y sus implementaciones clásicas. Comparación de las implementaciones clásicas de tipos abstractos de datos lineales.

Medidas de tiempo de ejecución de programas. Análisis de tiempo de ejecución en caso promedio y peor caso. Análisis de tiempo de ejecución de algoritmos iterativos y recursivos. Resolución de ecuaciones de recurrencias. Tasas de crecimiento de funciones de tiempo de ejecución. Análisis empírico del tiempo de ejecución de programas.

Tipos abstractos de datos más avanzados. Los tipos abstractos de datos Conjunto y Diccionario. Diccionarios y su implementación. Árboles como estructuras de datos. Recorridos sobre árboles (inorder, preorder, postorder). Árboles binarios de búsqueda. AVLs y árboles 2-3. Comparación de las implementaciones clásicas de diccionarios de acuerdo a la eficiencia de sus operaciones.

El tipo abstracto de datos Grafo. Algoritmos de recorrido/visita de grafos: recorridos "primero en profundidad" y "primero a lo ancho". Implementaciones clásicas de grafos (matrices de adyacencias y listas de adyacencias). Grafos dirigidos y no dirigidos. Grafos con costos. Aplicaciones del tipo abstracto de datos Grafo. Algoritmos clásicos sobre grafos. Cálculo de camino de mínimo costo: Algoritmos de Dijkstra y Floyd. Clausura transitiva: Algoritmo de Warshall. El problema del árbol abarcador de costo mínimo: Algoritmos de Prim y Kruskal.

Estructuras de datos avanzadas. Heaps y sus aplicaciones. Diagramas binarios de decisión (BDDs) y aplicaciones. Hashing.

Cronograma de Clases y Parciales

Semana							
Fecha	Teóricos	Fecha	Prácticos	Fecha	Laboratorios	Parciales/Rec	
1	17/08/16	Introducción a la materia			18/08	repaso	
1	19/08/16	Mecanismos de Reuso					
2	24/08/16	Encapsulación			22/08	OO	
2	26/08/16	Prog. OO	25/08	OO			
3	31/08/16	Prog. OO y Ejemplos.			29/08	OO	
3	02/09/16	TADs y Algebras	01/09	OO			
4	07/09/16	Algebras			06/09	OO	
4	09/09/16	Funciones de Abstracción	08/09	Algebras			
5	14/09/16	Introd. Java			12/09	Imp.Algebras	
5	16/09/16	Introd. Java	15/09	Intro.Java			
6	21/09/16	Excepciones en Java			19/9	Java	
6	23/09/16	Analisis de tiempo de ejecución	22/9	Analisis de ej.			
7	28/09/16	Big Oh, Big Omega, etc			26/09	Analisis de ej.	

7	30/09/16	Sorting	29/09	Analisis de ej.			Primer Parcial (30 Nov.)
8	05/10/16	Sorting cont.			03/10	Impl.Sorting	
8	07/10/16	Arboles	6/10	Ana.Sorting			
9	12/10/16	Arboles (Cont.)			12/10	Ej. Arboles	
9	14/10/16	Diccionarios e implementaciones	13/10	Arboles			Rec. Primer Parcial (14/10)
10	19/10/16	Diccionarios e implementaciones			17/10	Imp.Diccionarios	
10	21/10/16	Grafos	20/10	Diccionarios			
11	26/10/16	Algoritmos sobre Grafos			24/10	Imp.Grafos	
11	28/10/16	Algoritmos sobre grafos	27/10	Grafos			
12	02/11/16	Intro. complejidad			31/10	Imp.Grafos	4 de Nov. Entrega TP
12	04/11/16	Intro. Complejidad	03/11	Alg.Grafos Repaso			
13	09/11/16	Estructuras avanzadas (binomial heaps, arboles B*, etc)			07/11 10/11	Imp.Alg.Grafos	Primer Parcial (7 Nov)
13	11/11/16	Estruc. Avanzadas					
14	16/11/16	Repaso					14 Nov. 2do rec.

Bibliografía Obligatoria:

Mark Allen Weiss, " Data Structures and Algorithm Analysis in *Java*", Addison-Wesley, 2010
T.Cormen, C. Leiserson, R.Rivest. C.Stein, "Introduction to Algorithms", The MIT Press, 2001.

Bibliografía de Consulta:

A. Aho, J. Hopcroft y J. Ullman, "Data Structures and Algorithms", Addison-Wesley, 1983.
U. Manber, "Introduction to Algorithms: A Creative Approach", Addison-Wesley, 1989.

- B. Meyer, "Object Oriented Software Construction", Prentice-Hall, 2000.
- B. Liskov, "Program Development in Java", Addison-Wesley, 2000.
- A. Levitin, "Introduction to the Design and Analysis of Algorithms", Addison-Wesley, 2003.