PROGRAMA de Estructuras de Datos

Carreras: Tecnicatura Universitaria en Programación Informática - Licenciatura en Informática - Licenciatura en Bioinformática

Asignatura: Estructuras de Datos

Núcleo al que pertenece: Básico

Profesores: Pablo E. Martínez López – Pablo Barenbaum

Asignaturas Correlativas: Introducción a la Programación

Objetivos:

Que quienes cursen la materia:

- Comprenda la noción de dato y de estructuras de datos, y su importancia e interrelación estrecha con la estructura algorítmica de un programa.
- Entienda la diferencia entre acceso aleatorio y acceso secuencial.
- Conozca la idea de interface de una estructura de datos, y sea capaz de utilizarla productivamente para la solución de problemas.
- Conozca la interface de distintas estructuras de datos básicas (pilas, colas, listas,árboles, hashing, etc.) y las utilice adecuadamente
- Comprenda y utilice la noción de estructura contenedora, y la capacidad de realizar combinaciones complejas utilizándolas (i.e. pila de lista de registros, etc.)
- Se familiarice con las nociones de ámbito y de pasaje de parámetros por valor o referencia.
- Maneje alguno de los principios básicos de diseño de interfases de una estructura de datos (separación en constructores e inspectores de una interfase, ecuaciones entre combinaciones de constructores, etc.), y pueda reconocerlos en situaciones prácticas junto con su utilidad.
- Comprenda el concepto de asignación dinámica de memoria, y pueda hacer programas que hagan un manejo dinámico explícito de memoria en forma adecuada.
- Entienda la noción de implementación de una estructura de datos, y de su eficiencia, y sea capaz de implementar las interfases vistas anteriormente con distintas alternativas variadas en eficiencia.
- Se familiarice con las tareas de compilar y vincular programas para lograr un ejecutable.

- Pueda resolver problemas mediante programas recursivos, y entienda la diferencia entre una resolución recursiva y otra iterativa.

Contenidos mínimos:

- Recursión sobre listas y árboles. Programas recursivos.
- Tipos algebraicos: maybe, either, enumerativos, listas, árboles binarios, árboles generales.
- Estructuras contenedoras: pilas, colas, diccionarios, heaps, árboles balanceados, contenedores basados en representaciones numéricas.
- Nociones de representación e invariante de representación y su utilidad en el diseño e implementación de estructuras de datos.
- Uso imperativo de estructuras de datos. Iteración en listas y árboles.
- Modelo de memoria imperativo: stack/heap, alocación de memoria. Punteros. Variables por referencia.
- Listas encadenadas y sus variantes. árboles implementados con punteros. Binary heaps implementadas con arrays.
- Hashing. Análisis de eficiencia e implementación.
- Algoritmos de ordenamiento. Clasificación e implementación.
- Nociones básicas de algoritmos sobre grafos.

Carga horaria semanal: 8 hs

Programa analítico:

Unidad 1: Introducción a Estructuras de Datos

Nociones básicas de estructuras de datos. Listas. Recursión. Pattern Matching. Tipos algebraicos: Maybe, Either, Enumerativos, Representaciones Numéricas. Árboles binarios y generales.

Unidad 2: Estructuras de datos en alto nivel (puro)

Pilas, Colas, Conjuntos. Nociones de representación e invariante de representación/precondición Maps. Implementaciones diversas: Listas de pares clave-valor, BSTs, AVLs. Heaps. Binary heaps.

Unidad 3: Introducción a imperativo

Modelo de memoria imperativa, lenguaje C. Memoria Stack y memoria Heap. Alocación de memoria. Punteros. Variables por referencia. Arrays.

Unidad 4: Estructuras de datos en bajo nivel (imperativo/destructivo)

Linked lists (destructivas). Árboles con punteros. Iteración en listas y árboles (puros). Binary Heaps con arrays. Hashing.

Unidad 5: Temas adicionales

Sorting**.** Nociones básicas sobre grafos

Bibliografía obligatoria:

- Chris Okasaki, Purely Functional Data Structures. Cambridge University Press, 2009.
- Lipovaca, M. Learn you a haskell for great good!: A beginner's guide. No starch press. 1st Edition. 2011
- Barnett, G., & Del Tongo, L. Data Structures and Algorithms: Annotated Reference with Examples. DotNetSlackers. 2008
- Reema Thareja. Data Structures Using C. Oxford. 2014.
- Mark Allen Weiss , Data Structures and Algorithm in C. Addison-Wesley, 1997 (2da edición).

Bibliografía de consulta:

- Thomas Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein , Introduction to Algorithms. MIT Press, 2009.
- Peter A. Fejer y Dan A. Simovici, Mathematical Foundations of Computer Science. Vol.I: Sets, Relations, and Induction. Springer Verlag, 1991..

Organización de las clases:

Se realizarán clases teóricas (una vez por semana) y prácticas de programación en papel y computadora (dos veces por semana).

TP N°1:
Los objetivos del trabajo son: familiarizarse con el lenguaje Haskell; aprender a definir tipos algebraicos, y funciones a través del uso de pattern matching, polimorfismo paramétrico y recursión; aprender a definir funciones sobre listas.

TP N°2:
Los objetivos del trabajo son: aprender a definir tipos algebraicos arbóreos y funciones sobre los mismos.

TP N°3:
Los objetivos del trabajo son: aprender a definir tipos algebraicos en general, así como funciones sobre los mismos (registros, tipos algebraicos como Maybe, Either, etc.).

TP N°4:
Los objetivos del trabajo son: aprender a definir tipos abstractos simples, definir interfaces, aprender a calcular costos sobre funciones y comprender el uso de tipos abstractos cómo mecanismo de reutilización de código.

TP N°5:
Los objetivos del trabajo son: aprender a definir distintas variantes de pilas y colas como tipos abstractos y aprender algunos algoritmos que se sustentan sobre dichas estructuras.

TP N°6:
Los objetivos del trabajo son: aprender a definir diccionarios utilizando listas, así como su uso en diversos algoritmos.

TP N°7:
Los objetivos del trabajo son: aprender a definir diccionarios mediante árboles de búsqueda binaria, y aprender a definir algoritmos utilizando árboles de búsqueda en general. Además se verá cómo mejorar la eficiencia a través de la definición de AVLs.

TP N°8:
Los objetivos del trabajo son: aprender a colas de prioridad a través de binary heaps, así como algoritmos donde esta estructura está involucradas, como heapsort.

TP N°8:
Los objetivos del trabajo son: introducir el uso C++, aprender a resolver problemas utilizando estructuras, arrays y punteros. Modelar registros como tipos abstractos a través de módulos.

TP N°9:
Los objetivos del trabajo son: aprender a definir listas enlazadas utilizando punteros. Aprender ciertas mejoras sobre la definición de estas listas, mejorando la eficiencia de toda su interfaz. Aprender a definir iteradores sobre las mismas.

TP N°10:
Los objetivos del trabajo son: aprender a definir algoritmos imperativos sobre árboles binarios utilizando DFS y BFS.

TP N°11:

Los objetivos del trabajo son: aprender a definir árboles de búsqueda binaria y heaps utilizando arrays.

TP N°12:
Los objetivos del trabajo son: aprender a definir diccionarios utilizando tablas de hash. Aprender ciertos algoritmos clásicos de ordenamiento, principalmente sobre arrays.

Modalidad de evaluación:

Los mecanismos de evaluación en modalidades libre y presencial de esta asignatura están reglamentados según los siguientes artículos del Régimen de estudios de la UNQ (Res. CS 201/18).
En la modalidad de libre, se evaluarán los contenidos de la asignatura con un examen escrito, un examen oral e instancias de evaluación similares a las realizadas en la modalidad presencial.

CRONOGRAMA TENTATIVO

Semana	Tema/unidad	Actividad*				Evaluación
		Teórico	Práctico			
			Res Prob.	Lab.	Otros Especificar	
1	Presentación de Haskell y Hugs, Listas, Pattern Matching	x	x	x		
2	Tipos Algebraicos, Árboles Binarios	x	x	x		
3	Arboles Binarios, Expresiones Aritméticas	x	x	x		
4	Parcial Tipos Algebraicos, Invariantes de Representación, Tipos Abstractos de Datos, Colas, Pilas, Conjuntos	x	x	x		x
5	Colas y otros Tipos Abstractos de Datos	x	x	x		
6	Diccionarios	x	x	x		
7	Diccionarios, BSTs, AVLs	x	x	x		
8	Heaps	x	x	x		
9	Repaso, Parcial Tipos Abstractos de Datos	x	x	x		x
10	Modelo de memoria, Punteros, Pasaje por Referencia, Representación de datos en memoria	x	x	x		
11	Listas destructivas y funcionales, recorridos iterativos y recursivos	x	x	x		
12	Tipos Abstractos de Datos con interfaz destructiva, pilas, colas, conjuntos	x	x	x		
13	Arrays, Listas con iteradores, recorridos con iteradores	x	x	x		
14	Arboles Binarios destructivos, recorridos iterativos sobre árboles, DFS, BFS	x	x	x		
15	Heaps, Hashing	x	x	x		
16	Sorting y Grafos	x	x	x		
17	Parcial Modelo Imperativo	x	x	x		x
18	Recuperatorios	x	x	x		x
19	Integrador	x	x	x		x

Universidad Nacional de Quilmes

Firmado digitalmente por: SUAREZ Mariana Alejandra
Directora del Departamento de Ciencia y Tecnologla
UNIVERSIDAD NACIONAL DE QUILMES