UNIVERSIDAD SIMÓN BOLÍVAR
Vicerrectorado Académico

1. Departamento: *Computación y Tecnología de la Información (6510)*

2. **Asignatura: Algoritmos y Estructuras II**

3. Código de la asignatura: CI2612

No. de unidades-crédito: 3

No. de horas semanales: Teoría 3 Práctica 1 Laboratorio 0

4. Fecha de entrada en vigencia de este programa: Enero 2015

5. OBJETIVO GENERAL: *Al final del curso, el estudiante está capacitado para diseñar e implementar un Tipo Abstracto de Datos (TAD), aplicar algoritmos y estructuras de datos clásicos en la solución de problemas computacionales y analizar la complejidad de algoritmos sencillos.*

6. OBJETIVOS ESPECÍFICOS: *El estudiante tendrá competencias para:*

1. *Interpretar el comportamiento de un TAD en el estilo de especificación basado en modelos*
2. *Especificar formalmente un TAD utilizando el estilo basado modelos descritos mediante la lógica de primer orden y estructuras matemáticas (conjuntos, multiconjuntos, secuencias, relaciones y funciones).*
3. *Utilizar técnicas de refinamiento de datos que permiten garantizar la consistencia de los modelos en la implementación de un TAD.*
4. *Seleccionar estructuras de datos para implementar un TAD utilizando criterios de eficiencia en tiempo y recursos.*
5. *Resolver problemas mediante el uso de algunos TADs (TAD Pila, TAD Cola, TAD Diccionario, entre otros) y algoritmos clásicos (algoritmos de Búsqueda y Ordenamiento).*
6. *Representar información con tipos de datos estructurados recursivos*
7. *Diseñar programas estructurados que incluyan datos y funciones recursivos*
8. *Aplicar las nociones básicas del análisis de algoritmos en casos sencillos.*

7. CONTENIDOS:

TEORÍA

1. Tipos Abstractos de Datos (TAD): Concepto. Especificaciones formales con modelos. Ejemplo Diccionario. Modelos abstractos de representación: conjuntos, multiconjuntos, secuencias, relaciones y funciones. Implementación: refinamiento de datos, modelo concreto, invariante de acoplamiento. Correctitud de refinamiento de datos. Encapsulamiento y ocultamiento. Módulos y clases. (6 horas)
2. Tipos Concretos de Datos: arreglos, registros, referencias o apuntadores. Implementación de tipos recursivos con estructuras de enlace simple y doble. Introducción a las tablas de Hash. (4 horas)
3. Tipos Algebraicos Libres: ejemplos (expresiones), definición de funciones, demostración de propiedades, primitivas de programación, programación de ejemplos iterativos y recursivos. (4 horas)
4. Conceptos de Análisis de Complejidad de Algoritmos (notación O grande, Θ grande y Ω grande; manejo de recurrencias; Teorema Maestro). Aplicación de éstos conceptos en el análisis de algoritmos clásicos (2 horas)
5. Algoritmos de Búsqueda y Ordenamiento: búsqueda secuencial, búsqueda binaria, ordenamiento por inserción, ordenamiento por selección, ordenamiento por intercambio (burbuja), ordenamiento por mezcla, ordenamiento rápido (Quicksort), ordenamiento de montículo (Heapsort). (4 horas)
6. Árboles. Árboles Binarios de Búsqueda: definición, propiedades, búsqueda, inserción y eliminación. Implementación. (6 horas)
7. Recursión vs. Iteración. Ejemplos sobre árboles. Uso de estructuras auxiliares (pila y cola) para implementar algoritmos iterativos. (4 horas)

PRÁCTICA

1. Ejercicios sobre especificación de TADs (Pila y Cola) (2 Horas).
2. Ejercicios sobre implementación de TADs (2 Horas)
3. Ejercicios sobre tipos algebraicos libres (2 Horas)
4. Ejercicios sobre implementación de algoritmos iterativos de búsqueda y ordenamiento (1 Hora)
5. Ejercicios sobre implementación de algoritmos recursivos de búsqueda y ordenamiento (1 Hora)
6. Ejercicios sobre árboles binarios de búsqueda (2 Horas)
7. Ejercicios sobre recursión vs. iteración (2 Horas)

8. ESTRATEGIAS METODOLÓGICAS, DIDÁCTICAS O DE DESARROLLO DE LA ASIGNATURA:

1. *Sesiones de teoría impartidas mediante clases magistrales.*
2. *Sesiones de ejercicios y/o problemas con discusión grupal.*
3. *Trabajos en grupo con ejercicios a resolver fuera del aula. Las dudas sobre estas tareas se*

aclaran en horas de consulta.

A lo largo del curso se presenta la noción de TADs. En la primera parte se hace énfasis en la especificación formal con modelos. Luego se dan varias implementaciones de los TADs con tipos concretos de datos, mostrando las bondades de unas respecto a las otras en base a criterios de eficiencia y uso de los recursos. Posteriormente, se presentan las estructuras de datos dinámicas como implementaciones alternativas. Luego se introducen los algoritmos clásicos de búsqueda y ordenamiento, su utilidad en la solución de una amplia variedad de problemas computacionales, así como su aplicación en muchos TADs. Se realiza el análisis de complejidad de estos algoritmos.

9. ESTRATEGIAS DE EVALUACIÓN:

1. *Se sugiere realizar tres pruebas escritas con un porcentaje de 30% cada una. Estas pruebas son parciales. Se hacen ejercicios de: completar la especificación formal de un TAD, completar la implementación de un TAD, corrección de refinamiento de datos, diseño de algoritmos recursivos, diseño de tipos algebraicos libres, uso e implementación de estructuras de datos dinámicas, completar la implementación de algoritmos de búsqueda y ordenamiento, análisis de complejidad .*
2. *Se sugiere enviar ejercicios, tareas o asignaciones para fuera del aula con un porcentaje del 10%.*

10. FUENTES DE INFORMACIÓN

Libro de Texto:

1. *T.H. Cormen, C.E. Leiserson, R.L. Rivest & C. Stein, Introduction to Algorithms, The MIT Press, 3ra. edición, 2009. Capítulos 1, 2, 3 y 4; 6 y 7; 10, 11, 12 y 13; Apéndice A.*

Guías complementarias:

2. *J.N. Ravelo, Especificación e Implementación de Tipos Abstractos de Datos, Universidad Simón Bolívar, 2012, disponible en ttp://www.ldc.usb.ve/˜jravelo/docencia/algoritmos/material.*
3. *J.N. Ravelo & K. Fernández, Tipos Algebraico-Libres, Universidad Simón Bolívar, 2012, disponible en http://www.ldc.usb.ve/˜jravelo/docencia/algoritmos/material.*
4. *J.N. Ravelo & K. Fernández, Ordenamiento sobre Arreglos, Universidad Simón Bolívar, 2012, disponible en http://www.ldc.usb.ve/˜jravelo/docencia/algoritmos/material.*

Otras referencias:

5. *N. Wirth, Algorithms + Data Structures = Programs, Prentice Hall, 1976. Capítulos 1 y 4.*
6. *B. Liskov con J. Guttag, Program Development in Java – Abstraction, Specification, and Object-Oriented Design, Addison-Wesley, 2001. Capítulos 1 al 10.*
7. *R. Mitchell, Abstract Data Types and Modula-2, Prentice Hall, 1992. Capítulos 1, 2 y 4.*
8. *C. Morgan, Programming from Specifications, Prentice Hall, 2da. edición, 1994. Cap. 9.*
9. *S. Thompson, Haskell – The Craft of Functional Programming, Addison-Wesley, 1996. Cap.10.*
10. *A.V. Aho, J.E. Hopcroft, J.D. Ullman, Data Structures and Algorithms, Addison-Wesley, 1983.*

11. CRONOGRAMA DE ACTIVIDADES

Esta sección es un apéndice a ser desarrollado por el profesor al inicio de cada ejecución del programa, y que debe informarse a los estudiantes). Éste orienta al estudiante y al docente sobre el desarrollo de la asignatura en el tiempo. El cronograma puede ser flexible y depende entre otros factores, del período de actividades docentes.

Se presenta el siguiente cronograma como modelo, el cual pretende enfatizar las horas de práctica, ya que no se tiene un día específico para las mismas.

Semana	Día 1	Día 2
1	TEMA 1	TEMA 1 (continuación)
2	TEMA 1 (continuación)	PRÁTICA 1
3	TEMA 2	TEMA 2 (continuación)
4	PRÁCTICA 2	PRUEBA ESCRITA 1
5	TEMA 3	TEMA 3 (continuación)
6	PRÁCTICA 3	TEMA 4
7	TEMA 5	PRUEBA ESCRITA 2
8	TEMA 5 (continuación)	PRÁCTICA 4 Y 5
9	TEMA 6	TEMA 6 (continuación)
10	TEMA 6 (continuación)	PRÁCTICA 6
11	TEMA 7	TEMA 7 (continuación)
12	PRÁCTICA 7	PRUEBA ESCRITA 3