UNIVERSIDAD SIMÓN BOLÍVAR
Vicerrectorado Académico

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

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

3. Código de la asignatura: CI2611

No. de unidades-crédito: 3

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

4. Fecha de entrada en vigencia de este programa: Abril 2012

5. OBJETIVO GENERAL: *Al final del curso, el estudiante está capacitado para especificar formalmente un problema computacional sencillo y diseñar una solución algorítmica correcta bajo el enfoque de programación estructurada*

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

1. *Interpretar un algoritmo especificado en el formalismo GCL*
2. *Especificar las entradas y salidas de un problema usando el lenguaje de la lógica*
3. *Hacer el análisis descendente de un problema computacional*
4. *Concebir una solución algorítmica para un problema computacional sencillo*
5. *Escribir algoritmos usando el formalismo GCL*
6. *Representar información con tipos de datos básicos, enumerados, secuenciales y estructurados no recursivos*
7. *Diseñar programas estructurados que incluyan datos y funciones no recursivos*
8. *Procesar datos contenidos en archivos de texto*
9. *Verificar formalmente la validez de un algoritmo que incluya datos y funciones no recursivos*

7. CONTENIDOS:

TEORÍA

Definición de Algoritmo. Especificación, Programación y Corrección de programas. (2 Horas)

Noción de Conjuntos, Secuencia y Funciones de Agregación (sumatoria, productoria, máximo, mínimo, conteo). Noción de constante y variable. Tipos Básicos (entero, caracter, real, boolean).

Especificaciones de entradas y salidas, precondición y postcondición. (2 Horas)

Estructura general de un programa en GCL. Tipos subrango y enumerados. Expresiones aritméticas y lógicas, precedencia. Asignación, instrucción nula (skip) y secuenciación. (2 Horas)

Estructuras de control: selección e iteración (2 Horas)

Corrección de programas: Tripletas Hoare, reglas generales, asignación, skip, secuenciación y selección (2 Horas).

Corrección de programas: invariantes, función de cota, teorema de Invarianza, iteración (2 Horas).

Corrección de programas: Búsqueda intuitiva de invariantes. (2 Horas)

Análisis Descendente. Subprogramas (funciones y procedimientos). Pasaje de parámetros. (2 Horas)

Arreglos. Uso de la iteración con arreglos. Arreglos Multidimesionales. Iteraciones Anidadas. (2 Horas)

Corrección de programas: Técnicas de derivación de invariantes: eliminación de un predicado de una conjunción, reemplazo de constantes por variables, fortalecimiento de invariantes. (3 Horas)

Corrección de programas: regla de llamada a procedimiento, regla de llamada a función, regla de modificación de arreglos. (2 Horas)

Constructor de tipos estructurados: record. Definición de nuevos tipos: type. Arreglos de estructuras (1 Horas)

Archivos secuenciales (2 Horas)

PRÁCTICA

Ejercicios sobre especificación de programas (2 Horas).

Ejercicios sobre construcción de algoritmos (2 Horas)

Ejercicios sobre prueba de corrección de algoritmos (2 Horas)

Ejercicios sobre búsqueda de invariantes y función de cota (2 Horas)

Ejercicios sobre análisis descendente, subprogramas y pasaje de parámetros (2 Horas)

Ejercicios sobre arreglos (2 Horas)

Ejercicios sobre técnicas de derivación de invariantes, reglas de corrección de llamadas a subprogramas y modificación de arreglos (2 Horas)

Ejercicios sobre tipos estructuras y archivos secuenciales (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 (programación y corrida en frío).*
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 ejercitan las nociones: especificación formal de problemas y diseño descendente de programas. En la primera parte se determinan especificaciones de problemas y se

desarrollan los algoritmos de manera intuitiva a partir de estas especificaciones, haciendo énfasis en esquemas de algoritmos y diseño descendente. Se dan varias soluciones algorítmicas a un mismo problema, mostrando las bondades de unas respecto a las otras y resaltando el buen estilo de la programación. La razón de dar pruebas de corrección formal de programas es para mostrar que existe una forma sistemática de desarrollar programas correctos a partir de una especificación y concientizar al estudiante de su responsabilidad de garantizar la corrección de un programa.

9. ESTRATEGIAS DE EVALUACIÓN:

1. *Tres pruebas escritas con un porcentaje de 30% cada una. Estas pruebas son parciales. Se hacen ejercicios de: especificación formal de problemas, completación de algoritmos en GCL, diseño de algoritmos de problemas sencillo usando GCL, demostraciones de la corrección de un algoritmo.*
2. *Ejercicios, tareas o asignaciones para fuera del aula con un porcentaje del 10%. Cada semana se manda una tarea con un número entre 5-10 ejercicios. De éstas se seleccionan dos y de cada una de ellas sólo se corrigen dos ejercicios.*

10. FUENTES DE INFORMACIÓN

1. *Oscar Meza. Introducción a la Programación. 2000. Disponible en la web:* http://ldc.usb.ve/~meza/ci-2615/
2. *Kaldewaij Anne. "Programming: the derivation of algorithms". Prentice Hall. 1990. ISBN- 0-13-204108-1. Capítulos 1, 2, 3 y 4.*
3. *Gries David. "The Science of Programming". Springer.Verlag.1981. ISBN 0-387-96480-0. Páginas 1-85 y 310-319*
4. *Gries David, Schneider Fred. "A Logical Approach to Discrete Math". Springer-Verlag. 1993.*
5. *Jesús Ravelo. Ejemplos de Especificación de Problemas Algorítmicos. 2009. Disponible en la web:* http://ldc.usb.ve/~jravelo/docencia/algoritmos/material/especs.pdf

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	TEORÍA: Definición de Algoritmo. Especificación, Programación y Corrección de programas	TEORÍA: Noción de Conjuntos, Secuencia y Funciones de Agregación (sumatoria, productoria, máximo, mínimo, conteo). Noción de constante y variable. Tipos Básicos (entero, caracter, real, boolean). Definición de nuevos tipos: type. Especificaciones de entradas y salidas, precondición y postcondición.
2	PRÁCTICA: Ejercicios sobre especificación de programas	TEORÍA: Estructura general de un programa en GCL. Tipos subrango y enumerados. Expresiones aritméticas y lógicas, precedencia. Asignación, instrucción nula (skip) y secuenciación
3	TEORÍA: Estructuras de control: selección e iteración. Arreglos. Uso de la iteración con arreglos.	PRÁCTICA: Ejercicios sobre construcción de algoritmos
4	PRUEBA ESCRITA 1	TEORÍA: Corrección de programas: Tripletas Hoare, reglas generales, asignación, skip, secuenciación y selección
5	TEORÍA: Corrección de programas: invariantes, función de cota, teorema de Invarianza, iteración.	PRÁCTICA: Ejercicios sobre prueba de corrección de algoritmos
6	TEORÍA: Corrección de programas: Búsqueda intuitiva de invariantes	PRÁCTICA: Ejercicios sobre búsqueda de invariantes y función de cota
7	TEORÍA: Análisis Descendente. Subprogramas (funciones y procedimientos). Pasaje de parámetros	PRÁCTICA: Ejercicios sobre análisis descendente, subprogramas y pasaje de parámetros
8	PRUEBA ESCRITA 2	TEORÍA: Arreglos Multidimesionales. Iteraciones Anidadas. Constructor de tipos estructurados: record. Arreglos de estructuras
9	PRÁCTICA: Ejercicios sobre arreglos multidimensionales y registros	TEORÍA: Corrección de programas: Técnicas de derivación de invariantes: eliminación de un predicado de una conjunción, reemplazo de constantes por variables.
10	TEORÍA: Corrección de programas: Técnicas de derivación de invariantes: fortalecimiento de invariantes, regla de llamada a procedimiento.	TEORÍA: Corrección de programas: regla de llamada a función, regla de modificación de arreglos.
11	PRÁCTICA: Ejercicios sobre técnicas de derivación de invariantes, reglas de corrección de llamadas a subprogramas y modificación de arreglos	TEORÍA: Archivos secuenciales
12	PRÁCTICA: Ejercicios sobre archivos secuenciales y repaso	PRUEBA ESCRITA 3