Universidad de Guadalajara
Centro universitario de los Altos
Licenciatura en

Programa de estudios por competencias
Seminario de solución de problemas de traductores de lenguajes II

1. Identificación del curso

Programa educativo: Ing. En Computación	**Unidad de aprendizaje: Seminario de solución de problemas de traductores de lenguajes II**	**Departamento de adscripción: Estudios Organizacionales**	
Academia: Sistemas digitales y de información	**Programa elaborado por:** María Obdulia González Fernández	**Modificado por:**	**Fecha elaboración/modificación: Enero 2016**

Clave de la asignatura:	**Horas teóricas:**	**Horas prácticas:**	**Total de Horas:**	**Créditos:**	**Tipo de materia**	**Área de formación:**	**Modalidad:**
	0	68	68	5	Curso		**Presencial**

Conocimientos previos:	**Unidad de aprendizaje precedente:**	**Unidad de aprendizaje subsecuente:**
Interpreta algoritmos computacionales y notaciones matemáticas (autómatas de estado finito y autómatas de pila), maneja la programación para la solución de aplicaciones, aplica las estructuras de datos en la solución de problemas, manipula las operaciones básicas de los archivos.	Seminario de solución de problemas de Traductores de lenguajes I Teoría de la Computación	Ninguno

2. Presentación

La presente unidad tiene la finalidad de desarrollar las capacidades necesarias para comprender las bases teóricas para la construcción e implementación de un compilador. Así como Comprender los algoritmos que se utilizan para resolver cada una de las fases del compilador. Al mismo tiempo el dominar completamente el proceso para convertir un lenguaje de alto nivel a un lenguaje de bajo nivel.

3. Competencia general (Unidad de competencia)

Programa las diferentes fases de un compilador comprendiendo las diferentes fases que atraviesa un lenguaje de alto nivel a un lenguaje de bajo nivel.

4. Elementos de competencia

A. Diseña y programa un analizador léxico de manera optima a partir de autómata de pila.

Requisitos		
Cognitivos: (Contenidos).	**Procedimentales:**	**Actitudinales:**
Analiza los elementos de un analizador léxico como: • Funciones del analizador léxico • Componentes: léxicos, patrones y lexemas • Determinación de los componentes léxicos mediante autómatas finitos. • Tabla de tokens • Errores léxicos • Generadores de analizadores Léxicos	Diseñar de un autómata finito para la construcción de un lenguaje de programación propio. Crear y programa la tabla de tokens. Distinguir los errores léxicos. Definir las reglas de un lenguaje de programación propio. Construir un analizador léxico en Java o en C++	• Acepta y respeta las opiniones de los demás. • Colabora con sus compañeros con la finalidad de mejorar el trabajo en equipo. • Muestra interés al aprendizaje continuo y autogestivo. • Valora la retroalimentación grupal.
Estrategias didácticas:	**Recursos requeridos**	**Tiempo estimado:**
Estrategias para indagar sobre conocimientos previos. Prácticas de laboratorio.	-Bibliografía básica -Antología -Cañón, laptop, plataforma virtual.	4 semanas

Exposición por parte del profesor. Investigación y participación de los alumnos	-Herramientas de Web 2.0	
Criterios de desempeño:	**Evidencias:**	**Producto esperado:**
-ejercicios resueltos a partir de operaciones de lenguajes, representación matemática de lenguajes y expresiones regulares. -Reportes de investigación de diversas fuentes de información.	Reportes de prácticas	Analizador léxico de un lenguaje propio.

B. Crea un analizador sintáctico a partir de autómata de estado finito definido que reconozca una gramática propia. (utilizando YACC o la librería JFlex de Java)		
Requisitos		
Cognitivos: (Contenidos).	**Procedimentales:**	**Actitudinales:**
Gramáticas libres de contexto Árboles de derivación. Análisis sintáctico descendente Análisis sintáctico ascendente Diagramas de sintaxis Eliminación de la ambigüedad. Manejo de errores	Identificar y crear una gramática formal Determinar la sintaxis de una gramática utilizándola notación la BNF (Backus-Naur Form). Construir diagramas de sintaxis de un lenguaje propio. Eliminar la ambigüedad de una gramática. Distinguir los Errores sintácticos. Programar un analizador sintáctico (utilizar un generador de analizador sintáctico como YACC o la librería JFlex de Java).	• Acepta y respeta las opiniones de sus compañeros. • Colabora con sus compañeros con la finalidad de mejorar el trabajo en equipo. • Muestra interés al aprendizaje continuo y autogestivo. • Valora la retroalimentación grupal.
Estrategias didácticas:	**Recursos requeridos**	**Tiempo estimado:**
Estrategias para indagar sobre conocimientos previos.	-Bibliografía básica Antología	6 semanas

Estrategias que promuevan la comprensión mediante la organización de información. Exposición por parte del profesor. Investigación y participación de los alumnos	-Cañón, laptop, plataforma virtual, software programación C++ o Java	
Criterios de desempeño:	**Evidencias:**	**Producto esperado:**
-ejercicios resueltos a partir de operaciones de lenguajes, representación matemática de lenguajes y expresiones regulares. -Reportes de investigación de diversas fuentes de información.	Reportes de prácticas	Analizador sintàctico

C. **Diseña y programa arboles de expresiones dirigida por la sintaxis un analizador semántico para un meta-compilador.**		
Requisitos		
Cognitivos: (Contenidos).	**Cognitivos: (Contenidos).**	**Cognitivos: (Contenidos).**
Arboles de expresiones y acciones semánticas de una analizador sintáctico. Tipos de analizadores semánticos Expresiones de tipos. Manejo de errores semánticos.	Detectar los errores semánticos. Establecer las reglas para la conversión de tipos en las expresiones. Actualizar la tabla de conversión de símbolos y de direcciones. Programar un analizador semántico.	• Acepta y respeta las opiniones de sus compañeros. • Colabora con sus compañeros con la finalidad de mejorar el trabajo en equipo. • Muestra interés al aprendizaje continuo y autogestivo. • Valora la retroalimentación grupal.
Estrategias didácticas:	**Recursos requeridos**	**Tiempo estimado:**
Estrategias para indagar sobre conocimientos previos. Estrategias que promuevan la comprensión mediante la organización de información.	-Bibliografía básica Antología -Cañón, laptop, plataforma virtual, software programación C++, Java	3 semanas

Exposición por parte del profesor. Investigación y participación de los alumnos		
Criterios de desempeño:	**Evidencias:**	**Producto esperado:**
-ejercicios -Reportes de investigación de diversas fuentes de información.	Prácticas de laboratorio	Analizador semántico

D. Genera una máquina virtual que ejecute código intermedio a partir del código fuente de un lenguaje propio.		
Requisitos		
Cognitivos: (Contenidos).	**Cognitivos: (Contenidos).**	**Cognitivos: (Contenidos).**
Introducción Notaciones Representaciones de código intermedio 1. Código P 2. Triplos 3. Cuádruplos Esquemas e generación Traducción de expresiones. Código ensamblador.	Traducir los tipos de notación para la conversión de expresiones: Infija, prefija y posfija. Representar expresiones mediante el código intermedio. Crear una maquina virtual para que ejecute el código intermedio de un lenguaje propio.	• Acepta y respeta las opiniones de sus compañeros. • Colabora con sus compañeros con la finalidad de mejorar el trabajo en equipo. • Muestra interés al aprendizaje continuo y autogestivo. • Valora la retroalimentación grupal.
Estrategias didácticas:	**Recursos requeridos**	**Tiempo estimado:**
Estrategias para indagar sobre conocimientos previos. Estrategias que promuevan la comprensión mediante la organización de información. Exposición por parte del profesor. Investigación y participación de los alumnos	-Bibliografía básica -Antología -Cañón, laptop, plataforma virtual, software programación C++, Java	3 semanas

Criterios de desempeño:	Evidencias:	Producto esperado:
-ejercicios resueltos -Reportes de investigación de diversas fuentes de información.	Prácticas de laboratorio y ejercicios	Programa de máquina virtual de lenguaje intermedio

E. Aplica diferentes tipos de optimización que permita eficiente el código intermedio propio.		
Requisitos		
Cognitivos: (Contenidos).	**Cognitivos: (Contenidos).**	**Cognitivos: (Contenidos).**
Tipos de optimizaciones: 1. Optimización de Mirilla. 2. A partir de expresiones locales 3. A partir de bucles 4. Optimización Global Generación de código óptimo para expresiones.	Practicar las diferentes técnicas para la optimización del código intermedio generado. Analiza códigos intermedios para proponer mejoras.	• Acepta y respeta las opiniones de sus compañeros. • Colabora con sus compañeros con la finalidad de mejorar el trabajo en equipo. • Muestra interés al aprendizaje continuo y autogestivo. • Valora la retroalimentación grupal.
Estrategias didácticas:	**Recursos requeridos**	**Tiempo estimado:**
Estrategias para indagar sobre conocimientos previos. Estrategias que promuevan la comprensión mediante la organización de información. Exposición por parte del profesor. Investigación y participación de los alumnos	-Bibliografía básica - Antología -Cañón, laptop, plataforma virtual, software programación C++, Java	3 semanas
Criterios de desempeño:	**Evidencias:**	**Producto esperado:**
-Ejercicios.	-Optimización de su código intermedio (propio).	Optimización de su código intermedio (propio).

-Reportes de investigación de diversas fuentes de información.		

b- Evaluación y acreditación

Área de conocimiento:
a) Reportes de investigación (20%)
Área de habilidades y destrezas:
a) Practicas individuales (35%)
b) Proyecto final (40%)
Área de actitud:
c) Participación 5%
(Podrá agregar cuantos elementos requiera en cada uno de los apartados de evaluación)

c- Bibliografía

Ruiz Catalán, Jacinto. (2010). Compiladores: teoría e implementación. Primera edición. México: Alfaomega.
Alfonseca Moreno, Manuel(2006) Compiladores e intérpretes : teoría y práctica. Madrid. Pearson
Cantú Treviño Thelma(2015) Teoría de autómatas un enfoque práctico. México. Pearson

d- Perfil docente

El docente de esta materia deberá ser un profesionista con formación en las áreas de la computación, comunicaciones o informática; capaz de motivar a la investigación y creación de conocimiento, con habilidades para transmitir sus conocimientos y enseñar de forma interactiva propiciando en los alumnos el auto-aprendizaje.

.

Vo.Bo Dr. Juan Jorge Rodríguez Bautista

Jefe del departamento

Vo.Bo Mtra. María OBdulia González Fernández

Presidente de Academia