UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Licenciatura en Ciencias de la Computación

Facultad de Ciencias

Programa de la asignatura

Denominación de la asignatura:

Compiladores

Clave:	Semestre:	Eje temático:	No. Créditos:
0817	7	Integración Teoría-Práctica	10

Carácter: Obligatoria	Horas		Horas por semana	Total de Horas
Tipo: Teórico-Práctica	Teoría:	Práctica:	7	112
	3	4		
Modalidad: Curso	Duración del programa: Semestral			

Asignatura con seriación obligatoria antecedente: Organización y Arquitectura de Computadoras; Autómatas y Lenguajes Formales

Asignatura con seriación obligatoria subsecuente: Ninguna

Asignatura con seriación indicativa antecedente: Análisis de Algoritmos; Lenguajes de Programación

Asignatura con seriación indicativa subsecuente: Ninguna

Objetivo general:

Reconocer los entornos en los que es conveniente construir un compilador; revisar las distintas organizaciones y etapas de un compilador, tanto las herramientas para la generación automática de las etapas del compilador susceptibles de ser generadas automáticamente, como las consideraciones principales a tomar en cuenta para aquellas etapas que se deben desarrollar *ad-hoc*.

Índice temático

Unidad	Temas	Horas	
		Teóricas	Prácticas
I	Introducción a lo que es un compilador	3	4
II	Análisis léxico	6	8
III	Análisis sintáctico	6	8
IV	Análisis dependiente del contexto	6	8
V	Representaciones intermedias	3	4
VI	La abstracción de procedimiento	3	4

VII	Estructura del código generado	3	4
VIII	Introducción a optimización de código	3	4
IX	Análisis de flujo de datos	3	4
X	Elección de instrucciones	3	4
XI	Programación de instrucciones	6	8
XII	Asignación de registros	3	4
	Total de horas:	**48**	**64**
	Suma total de horas:	**112**	

Contenido temático	
Unidad	**Tema**
I Introducción a lo que es un compilador	
I.1	Justificación.
I.2	Áreas de aplicación.
I.3	Principios fundamentales de compiladores.
I.4	Estructura de un compilador.
I.5	Vista de alto nivel de la compilación.
I.6	Propiedades deseables en un compilador.
I.7	Resumen y perspectiva.
II Análisis léxico	
II.1	En qué consiste.
II.2	Reconocimiento de palabras.
II.3	Expresiones regulares.
II.4	Construcción del analizador léxico a partir de expresiones regulares.
II.5	Implementación de analizadores léxicos.
II.6	Herramientas automáticas para la construcción de analizadores léxicos.
III Análisis sintáctico	
III.1	Introducción.
III.2	Sintaxis de expresiones.
III.3	Reconocimiento descendente.
III.4	Reconocimiento ascendente.
III.5	Construcción de tablas LR(1).
III.6	Aspectos prácticos.
III.7	Manipulación de gramáticas (optimización y reducción del tamaño de tablas).
IV Análisis dependiente del contexto	
IV.1	Introducción.
IV.2	Sistemas de tipos.
IV.3	Gramáticas de atributos.
IV.4	Traducción \emph{ad-hoc} dirigida por la sintaxis.
IV.5	Temas avanzados (inferencia de tipos y cambio de asociatividad).
V Representaciones intermedias	
V.1	Taxonomía de representaciones intermedias.
V.2	Conjuntos de representaciones intermedias gráficas.
V.3	Representaciones intermedias lineales.

V.4	Forma de asignación simple estática.
V.5	Transformación de valores a nombres.
V. 6	Tablas de símbolos.
VI La abstracción de procedimiento	
VI.1	Aspectos importantes.
VI.2	Abstracción de control de flujo.
VI.3	Espacio de nombres.
VI.4	Comunicación de valores entre procedimientos.
VI.5	Establecimiento de la direccionabilidad.
VI.6	Ligado estandarizado.
VI.7	Manejo de la memoria.
VII Estructura del código generado	
VII.1	Importancia del tema.
VII.2	Asignación de espacios de almacenamiento.
VII.3	Operadores aritméticos.
VII.4	Operadores booleanos y relacionales.
VII.5	Almacenamiento y acceso a arreglos.
VII.6	Cadenas de caracteres.
VII.7	Referencias a estructuras.
VII.8	Construcciones para el control del flujo.
VII.9	Llamadas a procedimientos.
VII.10	Implementación en lenguajes orientados a objetos.
VIII Introducción a optimización de código	
VIII.1	Aspectos generales.
VIII.2	Antecedentes.
VIII.3	Expresiones redundantes.
VIII.4	Alcance de la optimización: métodos locales, super-locales, por región, globales y de toda la aplicación.
VIII.5	Numeración de valores sobre regiones mayores al bloque básico.
VIII.6	Eliminación global de redundancia.
VIII.7	Temas avanzados: clonación y sustitución en línea..
IX Análisis de flujo de datos	
IX.1	Aspectos generales.
IX.2	Análisis de flujo de datos iterativo.
IX.3	Forma de asignación simple estática.
X Elección de instrucciones	
X.1	Importancia para la portabilidad.
X.2	Esquema simple de recorrido de árboles.
X.3	Selección de instrucciones mediante apareamiento de árboles.
X.4	Selección usando optimización con mirilla *(peephole).*
XI Programación de instrucciones	
XI.1	Aspectos importantes.
XI.2	El problema de la programación de instrucciones.

XI.3	Programación en listas.
XII Asignación de registros	
XII.1	Importancia.
XII.2	Aspectos subyacentes.
XII.3	Asignación y alojamiento locales de registros.
XII.4	Alojamiento y asignación cruzando fronteras de bloques básicos.
XII.5	Alojamiento y asignación globales.

Bibliografía básica:

1. Keith D. Cooper y Linda Torczon, *Engineering a Compiler,* Morgan Kaufmann, 2004.
2. Michael Lee Scott, *Programming Language Pragmatics*, Morgan-Kauffman Publishers, 2009.
3. Aho, Lam, Sethi y Ullman, *Compilers, Principles, Techniques and Tools, Second Edition*, Pearson Education Inc., 2007.

Bibliografía complementaria:

4. Aho, Lam, Sethi y Ullman, *Compilers, Principles, Techniques and Tools, Second Edition*, Pearson Education Inc., 2007.
5. Aho y Ullman, *The Theory of Parsing, Translation and Compiling, Vol. 1, Parsing*, Prentice-Hall, 1972.
6. Aho y Ullman, *The Theory of Parsing, Translation and Compiling, Vol. 2,* Prentice-Hall, 1973.
7. Duck Grune, Henri E. Bal, Ceriel J. H. Jacobs y Koen G. Langendoen, *Modern Compiler Design*, John Wiley y Sons, Ltd., 2000.
8. Michael Lee Scott, *Programming Language Pragmatics*, Morgan-Kauffman Publishers, 2009.
9. Reinhard Wilhelm y Dieter Maurer, *Compiler Desig*, Addison-Wesley, 1995.

Sugerencias didácticas:		**Métodos de evaluación:**	
Exposición oral	**(X)**	**Exámenes parciales**	**(X)**
Exposición audiovisual	**(X)**	**Examen final escrito**	**( )**
Ejercicios dentro de clase	**(X)**	**Trabajos y tareas fuera del aula**	**(X)**
Ejercicios fuera del aula	**( )**	**Prácticas de laboratorio**	**( )**
Seminarios	**(X)**	**Exposición de seminarios por los alumnos**	**( )**
Lecturas obligatorias	**(X)**	**Participación en clase**	**(X)**
Trabajo de investigación	**( )**	**Asistencia**	**( )**
Prácticas de taller o laboratorio	**(X)**	**Proyectos de programación**	**(X)**
Prácticas de campo	**( )**	**Proyecto final**	**( )**
		Seminario	**( )**
Otras: _____		**Otras:** _____	

Perfil profesiográfico:

Egresado preferentemente de la Licenciatura en Ciencias de la Computación o Matemático con especialidad en computación con amplia experiencia de programación. Es conveniente que posea un posgrado en la disciplina. Con experiencia docente.