



Nombre de la asignatura				Compiladores				Clave de la asignatura			
								C0108080			
Área de formación	Docencia frente a grupo según SATCA				Trabajo de Campo Supervisado según SATCA				Carácter de la asignatura		
	HCS	HPS	TH	C	HTCS	TH	C	TC			
Sustantiva Profesional	2	2	4	4	0	0	0	4	(X) Obligatoria		() Optativa

SERIACIÓN		
Explícita		Implícita
Asignaturas antecedentes	Asignaturas subsecuentes	Conocimientos previos
Programación I	Ninguna	

PROPÓSITO DE LA ASIGNATURA	
Construir aplicaciones que conviertan programas escritos en lenguajes de programación de alto nivel a código máquina aplicando las diferentes fases que componen un compilador.	
COMPETENCIAS A DESARROLLAR	
Genéricas	Específicas
Capacidad de análisis y síntesis. Compromiso ético. Capacidad de aplicar los conocimientos en la práctica. Resolución de problemas. Trabajo en equipo. Trabajo autónomo.	Desarrollar sistemas de software integrando tecnologías para la solución de problemas, automatizando los procesos operativos, flujo de información y toma de decisiones en las organizaciones con un enfoque sistémico bajo estándares internacionales.



UNIDAD No. 1	Generalidades	Horas estimadas para cada unidad
		6
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
1.1. Cadenas, alfabetos y lenguajes. 1.2. Grafos y digrafos. 1.3. Teoría de conjuntos. 1.4. Clasificación y características de los lenguajes de programación. 1.5. Definiciones (compilador, intérprete, clasificación de los compiladores, formatos de código máquina). 1.6. Máquinas multinivel y virtuales. 1.7. Estructura básica del compilador.	Explica los conceptos fundamentales sobre cadenas, alfabetos, lenguajes, grafos, dígrafos, clasificación de lenguajes de programación y, máquinas multinivel y virtuales. Resuelve problemas planteados en lenguaje natural y/o fórmulas matemáticas sobre operaciones con conjuntos. Explica la estructura básica del compilador y el proceso de compilación.	Mapa conceptual del tema. Cuadro sinóptico del proceso de compilación. Diagrama de bloques de la estructura del compilador. Examen escrito.

UNIDAD No. 2	Análisis léxico	Horas estimadas para cada unidad
		11
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
2.1. Definición del analizador léxico y sus funciones. 2.2. Expresiones y conjuntos regulares. 2.3. Máquinas de estados finitos. 2.4. Autómatas de estados finitos. 2.5. Diagrama y matriz de transiciones. 2.6. Herramientas para generar analizadores léxicos. 2.7. Algoritmos de reconocimiento.	Explica la estructura básica del analizador léxico y sus funciones. Traduce especificaciones de conjuntos de cadenas en lenguaje natural a expresiones regulares. Elabora matrices de transiciones a partir de expresiones regulares. Aplica los algoritmos de reconocimiento lexicográfico para implementación de analizadores léxicos.	Ejercicios y problemas resueltos. Analizador léxico implementado en equipos de conformidad con la especificación del producto final (Prototipo de un traductor, intérprete y/o máquina virtual). Examen escrito.



UNIDAD No. 3	Gramáticas	Horas estimadas para cada unidad
		12
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
3.1. Jerarquía de gramáticas de Noam Chomsky. 3.2. Notaciones BNF y EBNF. 3.3. Grafos sintácticos. 3.4. Problemas de las gramáticas (reductibilidad, ambigüedad y corrección).	Utiliza las gramáticas libres de contexto para la especificación formal de lenguajes de programación. Interpreta especificaciones de gramáticas en notaciones BNF y EBNF. Construye grafos sintácticos a partir de una especificación gramatical. Identifica y resuelve problemas de reductibilidad y ambigüedad en las especificaciones gramaticales.	Mapa conceptual sobre la jerarquía de gramáticas. Ejercicios y problemas resueltos. Examen escrito.

UNIDAD No. 4	Análisis sintáctico	Horas estimadas para cada unidad
		13
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
4.1. Definición del analizador sintáctico y sus funciones. 4.2. Derivaciones y árboles. 4.3. Análisis sintáctico ascendente (down-top). 4.4. Análisis sintáctico descendente recursivo (top-down).	Explica la estructura básica del analizador sintáctico y sus funciones. Utiliza los algoritmos de derivaciones y de árboles para reconocimiento sintáctico de expresiones. Aplica algoritmos de análisis sintáctico para implementación de analizadores sintácticos.	Ejercicios y problemas resueltos. Analizador sintáctico implementado en equipos de conformidad con la especificación del producto final (Prototipo de un traductor, intérprete y/o máquina virtual). Examen escrito.



UNIDAD No. 5	Fase de síntesis	Horas estimadas para cada unidad	
		11	
CONTENIDOS			
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje	
5.1. Funciones semánticas. 5.2. Síntesis de un paso. 5.3. Diseño de una máquina virtual. 5.4. Representaciones intermedias (RI). 5.4.1. Notación polaca y polaca inversa (RPN). 5.4.2. Tercetos, Tercetos indirectos. 5.4.3. Cuartetos. 5.4.4. Árboles sintácticos. 5.5. Rutinas semánticas (RS). 5.5.1. Rutinas Semánticas y su aplicación con RPN. 5.5.2. Rutinas Semánticas y cuartetos. 5.5.3. Descenso recursivo y rutinas semánticas. 5.6. Generación de código (GC) a partir de RI. 5.6.1. Algoritmos. 5.6.2. Tipos de código máquina. 5.6.3. GC en postfijo. 5.6.4. GC a partir de cuartetos.	Describe los conceptos relacionados con funciones semánticas. Explica el proceso de síntesis de un paso. Explica el proceso de diseño de una máquina virtual. Elabora representaciones intermedias de código objeto. Aplica los algoritmos de generación de código para implementación de generadores de código objeto.	Ejercicios y problemas resueltos. Analizador semántico y generador de código objeto de conformidad con la especificación del producto final (Prototipo de un traductor, intérprete y/o máquina virtual). Examen escrito.	



UNIDAD No. 6	Tabla de símbolos y manejo de memoria	Horas estimadas para cada unidad
		11
CONTENIDOS		
Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
6.1. Introducción. 6.2. Métodos de acceso. 6.3. Lenguajes con estructura de bloques (PASCAL, C). 6.4. Lenguajes con control de amplitud (MODULA, ADA). 6.5. Manejo de las estructuras de datos. 6.6. Asignación estática, dinámica y mixta. 6.7. Técnicas de recuperación de errores.	Explica los métodos de acceso a memoria de diferentes tipos de lenguajes, de representación y manipulación de datos estructurados en memoria y de asignación de memoria a datos de programas. Aplica los algoritmos para la implementación de la tabla de símbolos (TDS) de un compilador.	Ejercicios y problemas resueltos. Compilador implementado en equipos de conformidad con la especificación del producto final (Prototipo de un traductor, intérprete y/o máquina virtual). Examen escrito.

Contenidos procedimentales	Contenidos actitudinales
Analiza, diseña e implementa compiladores con base en especificaciones formales de lenguajes de programación y arquitecturas computacionales.	Responsabilidad en la entrega de sus trabajos. Honestidad en el manejo y presentación de la información. Diligente en el manejo de los datos. Disposición para trabajar en equipo. Rigor en el manejo de la información. Respeto a las propuestas de sus compañeros.



Metodología para la construcción del conocimiento	
Actividades de aprendizaje con el docente	Actividades de aprendizaje autónomo
Realización de ejercicios. Resolución de problemas. Exposición de temas. Implementación de algoritmos.	Desarrollo de un compilador. Solución de problemas. Realización de ejercicios.

Evidencias de desempeño		
Acreditación	Evaluación	Calificación
Cumplir con lo establecido en el Reglamento Escolar vigente. Entrega de evidencias de aprendizaje.	A final de cada unidad. Al final del curso.	10% Mapas conceptuales, cuadros sinópticos, diagramas de bloques, ejercicios resueltos. 10% Analizador Léxico. 10% Analizador sintáctico. 10% Analizador semántico y generador de código. 40% Proyecto final de compilador. 20% Exámenes escritos.



FUENTES DE APOYO Y CONSULTA	
BÁSICA	
<ol style="list-style-type: none">1. Aho, Lam, Sethi, Ullman. (2014). Compilers. Principles, techniques and tools. England: Pearson Education Limited.2. Campbell, B., Iyer, S., Akbal-Delibá , B. (2013). Introduction to compiler construction in a Java world. USA: CRC Press.3. Catalán, Ruiz, J. (2012). Compiladores. Teoría e implementación. España: RC Libros.4. Martínez, López, F. J. & L., Ramallo, Martínez, A. (2014). Teoría, diseño e implementación de compiladores de lenguajes. España: Editorial Ra-Ma.5. Texier, J. D. & Bermúdez, M. (2011). Compiladores - Un enfoque diferente. Sistema de escritura de traductores. España: Editorial Academia Española.	
COMPLEMENTARIAS	
<ol style="list-style-type: none">1. Aho, A. V., Sethi, R.; Ullman, J. D. (2008). Compiladores. Principios, técnicas y herramientas. México: Pearson Education. *2. Cooper, K., D., Torczon, L. (2012). Engineering a compiler. USA: ELSEVIER.3. Dave, P., H., Dave, H., B. (2012). Compilers: Principles and practice. Delhi: Pearson.4. Dos, Reiss, A. J. (2012). Compiler construction using Java, JavaCC, and Yacc. USA: IEEE Computer Society, Inc./John Wiley & Sons, Inc.5. Grune, D., Bal, H. E., Cerial, J. H. J., Koen, G. L. (2007). Diseño de compiladores modernos. España: McGraw-Hill/Interamericana de España, S. A. U. *6. Loudon, K. C. (2005). Construcción de compiladores. Principios y práctica. México: Ed. Thompson. *7. Teufel & Teufel. (1995). Compiladores. Conceptos fundamentales. México: Addison-Wesley Iberoamericana. *	

*La bibliografía con antigüedad mayor de cinco años contiene información relevante para el desarrollo de esta asignatura. Cabe destacar que son textos clásicos con ejemplos didácticos de fácil comprensión para el estudiante. Son difíciles de conseguir en el mercado, pero se encuentran en los catálogos de varias bibliotecas.

RESPONSABLE DEL DISEÑO	
Elaborado por	Pablo Payró Campos, Óscar Ovando Bautista, Carlos Arturo Custodio Izquierdo.
Fecha de elaboración	20 de diciembre de 2016.