UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

COORDINACIÓN DE FORMACIÓN BÁSICA
COORDINACIÓN DE FORMACIÓN PROFESIONAL Y VINCULACIÓN UNIVERSITARIA
PROGRAMA DE UNIDAD DE APRENDIZAJE

I. DATOS DE IDENTIFICACIÓN

1. Unidad Académica: FACULTAD DE CIENCIAS.
2. Programa (s) de estudio: Licenciatura en Ciencias Computacionales 3. Vigencia del plan:
4. Nombre de la Asignatura: Compiladores 5. Clave:
6. HC: 2 HL 2 HT 1 HPC ____ HCL ____ HE 2 CR 7
7. Etapa de formación a la que pertenece: Disciplinaria
8. Carácter de la Asignatura: Obligatoria X Optativa _____
9. Requisitos para cursar la asignatura: Teoría de autómatas

Formuló: Lic. Pedro Pérez
Dra. María Victoria Meza Kubo

Vo. Bo. Dr. Alberto Leopoldo Morán y Solares

Fecha: Agosto de 2016

Cargo: Subdirector

II. PROPÓSITO GENERAL DE LA UNIDAD DE APRENDIZAJE

El alumno hará un estudio comparativo de los principios, técnicas y herramientas para el desarrollo de cada una de las etapas (de análisis y síntesis) que se encuentran involucradas en el proceso de compilación. Dar al alumno una visión amplia sobre el uso y aplicación de los compiladores como solución a disitintos problemas reales. Mostrar al alumno la importancia del desarrollo de compiladores, como una solución a problemas o como una herramienta más en el caso de necesitar un compilador con características especiales para el desarrollo de proyectos futuros. Esta materia es obligatoria y se encuentra en la etapa disciplinaria.

Será requisito haber acreditado el curso de Teoría de autómatas.

III. COMPETENCIA (S) DE LA UNIDAD DE APRENDIZAJE

Desarrollar compiladores a través de emplear las diferentes fases que constituyen el proceso de traducción de un lenguaje de programación, con el fin de utilizarlos en aplicaciones reales, con actitud creativa.

IV. EVIDENCIA (S) DE DESEMPEÑO

Desarrolla un compilador que incluya: lenguaje y gramática independiente al contexto, la exposición del sistema y resultados obtenidos

V. DESARROLLO POR UNIDADES

Competencia

Identificar un problema real donde implementar un compilador, a través del entendimiento de los conceptos y fases del proceso de compilación, para resolver un problema de compilación, tomando una actitud creativa.

Contenido temático	**Duración**
1. Introducción a los compiladores	
1.1 Introducción	4 horas
1.2 Fases de un compilador	
1.3 Análisis léxico	
1.4 Análisis sintáctico o gramatical	
1.5 Análisis semántico	
1.6 Generador de código intermedio	
1.7 Optimización	
1.8 Generación de código	

Competencia

Implementar un analizador léxico considerando las funciones de un analizador, que permita recuperarse de errores, con actitud creativa.

Contenido temático	**Duración**
2. Análisis léxico	
1.1 Función del analizador léxico	4 horas
1.2 Componentes léxicos, patrones y lexemas	
1.3 Atributos de los componentes léxicos	
1.4 Manejo de “buffers” de entrada	
1.5 Especificación de componentes léxicos	
1.6 Expresiones regulares y autómatas finitos	
1.7 Reconocimientos de componentes léxicos	
1.8 Manejo de errores léxicos	

Competencia

Aplicar la construcción de tablas de símbolos, con la solución de ejercicios, como base para la construcción de traductores de manera creativa.

Contenido temático	**Duración**
3. Tabla de símbolos	
3.1 Propósito de la tabla de símbolos	4 horas
3.2 Atributos y estructura de datos para una tabla de símbolos	
3.3 Operaciones en la tabla de símbolos	
3.4 Eficacia de las tablas de símbolos	

Competencia

Diseñar máquinas abstractas, lenguajes generados por gramáticas formales y expresiones regulares, mediante la solución de ejercicios, para analizar las relaciones que guardan entre ellos, con creatividad e iniciativa.

Contenido temático	**Duración**
4. Análisis de sintaxis	
4.1 Propósito del analizador sintáctico	6 hrs.
4.2 Lenguajes y gramáticas	
4.3 Gramáticas independientes al contexto	
4.4 Análisis sintáctico descendente	
4.5 Análisis sintáctico ascendente	
4.6 Análisis sintáctico predictivo	
4.7 Análisis sintáctico de precedencia	
4.8 Problemas del análisis sintáctico	
4.9 Manejo de errores	

Competencia Implementar un traductor empleando las técnicas para la construcción de sistaxis, que permita comprender la construcción de compiladores, con creatividad y eficiencia.	
Contenido temático 5. Traducción dirigida por sintaxis 5.1 Definiciones dirigidas por sintaxis 5.2 Construcción de árboles sintácticos 5.3 Análisis de definiciones dirigidas por la sintaxis	**Duración** 4 horas.
Competencia Implementar un algoritmo empleando las técnicas de análisis semántico, que permita comprender la construcción de compiladores, con creatividad y eficiencia.	
Contenido temático 6. Análisis semántico 6.1. Introducción 6.2. Verificación estática 6.3. Especificación de un comprobador tipos simple	**Duración** 4 horas.
Competencia Implementar un generador de código intermedio, empleando diferentes técnicas de generación de código, para la construcción de un compilador con creatividad y eficiencia.	
Contenido temático **7. Generación de código** 7.1 Introducción 7.2 Preparación para la generación de código 7.3 Generador de código simple 7.4 Estrategias para la generación de código	**Duración** 6 horas.

IV. ESTRUCTURA DE LAS PRÁCTICAS

No. de Práctica	Competencia(s)	Descripción	Material de Apoyo	Duración
1 (Taller)	Proponer una solucióna una problemática real, a través del análisis de alternativas actuales para la solución, que impacte en el desarrollo de un sistema de traducción con creatividad e innovación.	Realizar una propuesta de proyecto en el cual se realice una descripción amplia de una problemática real, antecedentes de solución de la misma, justificación, objetivos generales y específicos, metas, metodología del trabajo, plan de trabajo (calendarización) y limitaciones del trabajo.	Notas de clase • Papel y lápiz • Internet • Procesador de palabras.	8 horas
2 (Laboratorio)	Diseñar de interfaces de software a través del uso de técnicas de diseño para el desarrollo de un compilador, con creatividad e innovación.	Realizar un bosquejo de la interface gráfica del compilador, en el cual se especifiquen las acciones que usuario podrá desarrollar y la forma de interactuar del usuario con el sistema.	• Papel y lápiz • Programa para el diseño de gráficos • Procesador de palabras.	6 horas
3 (Laboratorio)	Implementar un analizador léxico, empleando cualquier lenguaje de programación, para el procesamiento de lenguaje natural, con creatividad e innovación.	Determinar los conjuntos de terminales y no terminales que conforman la gramática. • Papel y lápiz. • Procesador de palabras. 20 hrs 4 Implementación de análisis léxico. Implementar un analizador léxico que tenga como entrada un programa en el lenguaje creado y una tabla de símbolos;	Papel, lápiz, pizarrón y plumones.	6 horas

		como salida tenga una cadena de tokens y la tabla de símbolos actualizada en base al análisis léxico.		
4 (Laboratorio/)	Implementar un analizador sintáctico, empleando cualquier lenguaje de programación, para el procesamiento de lenguaje natural, con creatividad e innovación.	Elaborar estudio de las diferentes técnicas de análisis sintáctico para así aplica el más adecuado a las necesidades de la solución a implementar por el alumno. Implementar la(s) técnica(s) de análisis sintáctico, que tendrá como entrada una cadena de tokens, la cual fue previamente generada por el analizador léxico, para así determinar si se acepta o nó la cadena. El analizador deberá ser capaz de recuperarse ante la presencia de errores y reportar de los mismos y su tipo al usuario.	• Papel y lápiz • Procesador de texto • Lenguaje de programación.	6 horas
5 (Laboratorio)	Implementar un analizador semántico empleando cualquier lenguaje de programación, para el procesamiento de lenguaje natural, con creatividad e innovación.	Implementar una analizador semántico o comprobador de tipos, el cual recibirá como entrada la cadena de tokens aceptada por el analizador sintáctico y deberá producir como salida la aceptación o negación de la cadena. En caso de encontrar errores éste deberá recuperarse para continuar así el análisis y presentar un listado de errores y el tipo de error encontrado	Papel y lápiz • Procesador de texto • Lenguaje de programación	6 horas

6 (Laboratorio)	Implementar un traductor dirigido por sintaxis, empleando cualquier lenguaje de programación, para el procesamiento de lenguaje natural, con creatividad e innovación.	Implementar el traductor dirigido por sintaxis para el lenguaje de entrada a lenguaje objeto, éste último puede ser cualquier lenguaje de programación conocido por el alumno. La traducción generada en lenguaje objeto no deberá contener errores para que pueda ser utilizado.	• Papel y lápiz • Procesador de texto • Lenguaje d	8 horas
7 (Taller)	Realizar un reporte escrito, empleando los lineamientos vistos en clase, que describa el desarrollo del proyecto, con creatividad y responsabilidad.	Elaborar reporte el cuál reflejará el resultado final del proyecto, metas alcanzadas, conclusiones y trabajo a futuro relacionado con el proyecto, Desarrollar una presentación final de 20 min en la cual se presentarán los resultados finales y el compilador desarrollado así como su funcionamiento.	• Papel y lápiz • Procesador de texto • Lenguaje d	8 horas

VII. METODOLOGÍA DE TRABAJO

- Al término de la unidad 1 de la asignatura el estudiante deberá realizar una propuesta de proyecto aplicando los sistemas traductores (compiladores) en la solución de una problemática real. En la propuesta el equipo debe presentar una descripción de la problemática a ser atacada, la justificación del uso de un compilador como parte de la solución de la problemática.
- Implementación de sistema propuesto y presentación final. Los equipos deberán implementar el sistema de traducción (compiladores) propuesto al inicio de periodo, para así mostrar al final los resultados obtenidos en base a lo implementado.
- Trabajo en equipo, el proyecto se elaborará en equipo de máximo 3 personas
  A lo largo del periodo se irá implementado la propuesta hecha en equipo.
- Realización de tareas y ejercicios extra clase.
  Durante el periodo se realizarán varios ejercicios extra clase para una mejor comprensión de los temas.
- Exámenes escritos y orales.
  Por medio de exámenes al final de cada unidad, el alumno podrá verificar su avance o deficiencias y de esta forma poner mayor atención en el tema, de igual forma se evalúa si los temas visto durante la unidad han sido evaluados.
- Participación en clase
  A lo largo del periodo se desarrollarán ejercicios explicados y dirigidos por el maestro, los cuales servirán de guía para las tareas y ejercicios extra clase a ser realizadas por el alumno.

VIII. CRITERIOS DE EVALUACIÓN

Criterio de calificación

■ Exámenes parciales	20%
■ Ejercicios/prácticas	20%
■ Proyecto final	50%
■ Presentación del proyecto	10%
TOTAL	100%

Criterio de acreditación

- Resolver 4 exámenes parciales en tiempo y forma.
- Cumplir con las prácticas de laboratorio en tiempo y forma.
- Cumplir con la presentación y reporte de un proyecto en tiempo y forma.
- Para la acreditación del curso se atenderá al Estatuto Escolar Vigente, artículos 70-71, por lo que el estudiante deberá contar un mínimo de 80% de asistencias en el periodo. Tener un mínimo aprobatorio de 60 en su calificación final

Criterio de evaluación

- Tanto para el caso de las tareas como el de los exámenes, ambos serán resueltos en clase posterior para retroalimentar el desarrollo del curso.
- Deberán entregarse el 70% de prácticas de laboratorio para tener derecho a examen final.
- En el caso del proyecto final por equipo, la evaluación se dividirá en dos: reporte y desarrollo, en el primer caso los puntos a evaluar serán, contenido, limpieza, así como ortografía; para el segundo caso los puntos a evaluar serán, número de prácticas, fácil de utilizar e integración de las prácticas.
- Se busca con el proyecto en equipo formar valores de responsabilidad, búsqueda de la calidad, sentido de justicia, así como valores de síntesis y actracción.

IX. BIBLIOGRAFÍA

Básica	Complementaria
• Aho, A.V., Lam, Monica S, Sethi, R., Ullman, Jeffrey D. (2013). Compilers: Principles, Techniques, and Tools. Pearson. • Campbell, B., Iyer, S., & Akbal-Delibas, B. (2012). *Introduction to compiler construction in a Java world*. CRC Press. • Flex, Bison & MinGw (2016). Construcción de compiladores básicos. Editorial Académica Española.	• Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2006). Automata theory, languages, and computation. International Edition, 24. [clásico] • Recursos digitales: https://msdn.microsoft.com/en-us/magazine/dn904673.aspx • Recursos digitales: https://visualstudiomagazine.com/articles/2014/05/01/how-to-write-your-own-compiler-part-1.aspx

X. PERFIL DOCENTE

Profesionista en computación o área afín con experiencia docente y conocimientos en el manejo de compiladores y teoría de autómatas.