



Facultad de Ingeniería - Universidad Nacional de Cuyo			
P1- PROGRAMA DE ASIGNATURA			
Adecuación a las modalidades presencial y a distancia por Pandemia COVID-19			
Asignatura:	COMPILADORES		
Docente Responsable:	PROFESOR TITULAR, DRA. ANA CAROLINA OLIVERA		
Carrera:	Licenciatura en Ciencias de la Computación		
Año: 2020	Semestre: 7MO	Horas Semestre: 96	Horas Semana: 6

OBJETIVOS

- Aplicar los conceptos de teoría de lenguajes formales, ingeniería de software y programación orientada a objetos para el diseño y desarrollo de algoritmos de análisis léxico, sintáctico y semántico. Generación de código y optimización.
- Aplicar los conocimientos sobre Gramáticas Formales en la descripción de los lenguajes de programación, con el propósito de desarrollar parte de un compilador.
 - Aplicar conceptos de atributos heredados y sintetizados en la construcción de sistemas que requieran la aplicación de analizadores sintácticos.
 - Diferenciar las características propias de un compilador en comparación a un intérprete.

CONTENIDOS

UNIDAD 1: INTRODUCCIÓN Y ARQUITECTURA DE UN COMPILADOR

1.A Introducción A Compiladores E Intérpretes

Definición de traductor, intérprete y compilador. Problemas que debe resolver un compilador. Clasificación de los compiladores. Historia y evolución de los compiladores. Programas relacionados. Proceso de traducción. Relación entre compiladores, lenguajes formales y teoría de autómatas. Aplicaciones.

1.B Arquitectura De Un Compilador

Visión esquemática de un compilador. Front End y Back End de un Compilador. Definición dirigida por la sintaxis. Fases de un compilador.

UNIDAD 2: ANÁLISIS LÉXICO

2.A Diseño De Un Analizador Lexico

Qué es un analizador léxico. Diseño de analizadores léxicos, estructura del analizador. Interacción con la Tabla de Símbolos. Errores Léxicos. Construcción de un analizador léxico.

2.B Generación Automática De Analizadores Léxicos

Generadores de analizadores léxicos. SableCC, JavaCC.

UNIDAD 3: ANÁLISIS SINTÁCTICO

3.A Diseño De Un Analizador Léxico



Qué es un analizador sintáctico, representación de gramáticas, manejo de errores. Gramáticas libres de contexto, definiciones formales, derivaciones, árboles sintácticos y derivaciones, eliminación de ambigüedad, eliminación de recursividad.

3.B Análisis Sintáctico Descendente

Análisis de descenso recursivo, primero y siguiente. Análisis sintáctico predictivo no recursivo. Recuperación de errores. Análisis LL(1). Gramáticas ambiguas. Errores sintácticos.

3.C Análisis Sintáctico Ascendente

Análisis Sintáctico Ascendente. Handle. Ítems y clausura. Analizadores LR, construcción de tablas. Analizadores LR canónico, SLR y LALR. Desplazamiento-Reducción. Precedencia y Asociatividad.

3.D Generadores Automáticos De Analizadores Sintácticos

Generadores de analizadores sintácticos SableCC, JavaCC.

UNIDAD 4: ANÁLISIS SEMÁNTICO

4.A Gramática de Atributos

Gramáticas libres de contexto aumentadas. Traducción dirigida por la sintaxis, definiciones dirigidas por la sintaxis. Esquemas de traducción. Atributos heredados y sintetizados. Evaluación de reglas semánticas. Evaluación de atributos. Definiciones de atributos por la izquierda. Árbol de dependencia y orden topológico.

4.B Chequeo de Declaraciones

Tipos. Conversiones de tipo. Sobrecarga de funciones y de operadores. Polimorfismo. Diseño de la Tabla de Símbolos. Chequeo de Declaraciones. Chequeo de Declaraciones en el Paradigma orientado a objetos. Errores Semánticos asociados a las declaraciones.

4.C Chequeo de Sentencias

Árboles Sintácticos Abstractos. Chequeo de Sentencias. Chequeo de sentencias en el Paradigma orientado a objetos. Polimorfismo. Sobrecarga. Inferencia de tipos. Errores Semánticos asociados a las sentencias.

UNIDAD 5: AMBIENTES DE EJECUCIÓN Y GENERACIÓN DE CÓDIGO INTERMEDIO

5.A Tiempo de Ejecución

Organización de la memoria. Asignación de memoria. Tipos de Datos. Tabla de símbolos, organización y acceso. Activaciones. Árbol de Activación. Particularidades del Paradigma Orientado a Objetos. Alineación de los datos.

5.B Generación de Código Intermedio.

Aspectos del lenguaje fuente. Lenguajes intermedios. Máquina con Pila. Máquina con Registros. Uso de definiciones dirigidas por la sintaxis para la generación del código intermedio. Distintos tipos de representación. Código intermedio para una máquina virtual y código intermedio para un generador de código. La máquina objeto. Administración de la memoria en tiempo de ejecución. Uso de los registros rápidos de la máquina destino. Construcción de un generador de código simple. Optimización.

UNIDAD 6: OPTIMIZACIÓN DE CÓDIGO Y VALIDACIÓN DE UN COMPILADOR

6.A Optimización

Significado de optimización. Costo. Principales fuentes para la optimización. Tipos de optimización. Ejemplos de optimización.

6.B Validación de un compilador



Conceptos. Construcción de casos de prueba.

UNIDAD 7: INTÉRPRETES

7.A Interpretés Características Principales

Estructura de un intérprete. Tipos de Intérpretes. El rol de la Tabla de Símbolos en un intérprete. Construcción de un intérprete. Aplicaciones.

METODOLOGÍA DE ENSEÑANZA

La materia se organiza en clases teóricas, clases prácticas y el desarrollo por parte de cada alumno de un proyecto Front-End de un lenguaje de programación académico al que denominamos tinyJava.

En el contexto de pandemia COVID 19, la modalidad a distancia adoptada utiliza la plataforma de Aula Abierta para el desarrollo de las clases teóricas en vivo en los horarios habituales de cursada sumado a herramientas como diapositivas (PowerPoint) para ayudar al alumno en el seguimiento de la clase.

DISTRIBUCIÓN DE LA CARGA HORARIA

Actividad	Carga horaria por semestre
Teoría y resolución de ejercicios simples	50
Formación práctica	
Formación Experimental – Laboratorio	0
Formación Experimental - Trabajo de campo	0
Resolución de problemas de ingeniería	15
Proyecto y diseño	31
Total	96

Porcentaje de Horas Presenciales	15 % del Total
Porcentaje de Horas a Distancia	85 % del Total

BIBLIOGRAFÍA

Bibliografía básica

Autor	Título	Editorial	Año	Ejemplares en biblioteca
KENNETH, L.	Construcción de compiladores, principios y práctica.	Thomson.	2005	
AHO, A. et. al	Compiladores. Principios, técnicas y herramientas	Pearson Educación.	2008	
Cooper, K. y Torczon, L.	Engineering: A Compiler.	Morgan Kaufmann	2011	
APPEL, A. W. y Palsberg, J.	Modern Compiler Implementation in Java	Cambridge	2002	



		University Press		
--	--	---------------------	--	--

Bibliografía complementaria

Autor	Título	Editorial	Año	Ejemplares en biblioteca

EVALUACIONES (S/ Ord. 108-10_CS)

Se propone una evaluación continua de alumno de acuerdo al desarrollo de un proyecto de compilador. Se presenta a continuación las normas para regularizar y aprobar la materia.

Evaluaciones durante el cursado:

- Se realizará un proyecto de software que tiene como objetivo construir el Front-End de un compilador de un lenguaje académico orientado a objeto. El alumno debe aprobar el proyecto de Compilador y cada una de sus etapas con nota aprobado o su correspondiente re-entregas. El proyecto de compilador posee 5 (CINCO) etapas obligatorias durante el periodo lectivo. La etapa 5 además deberá responder preguntas orales para demostrar el conocimiento teórico de los tópicos abarcados por el proyecto de compilador.

Evaluación Recuperatoria y Global:

- Cada etapa tendrá su correspondiente instancia de recuperación a través de la reentrega de la misma en caso de obtener nota desaprobado en una entrega. Los archivos y su documentación quedarán a respaldo documental.

Condición de Regularidad

En base a los resultados de las evaluaciones el alumno quedará como Alumno Regular, cuando haya aprobado cada entrega o re-entrega del proyecto.

Alumnos recursantes. No hay régimen especial para alumnos recursantes.

Criterios de evaluación:

En cuanto a alumnos regulares: se valorará la precisión del alumno en las respuestas a las preguntas realizadas durante el examen oral. La coherencia de sus respuestas entre sí junto con el grado de conocimiento demostrado en la capacidad de relacionar los conceptos vistos durante la cursada. La madurez conceptual evidenciada en la capacidad de respuestas rápidas concisas y precisas a consultas realizadas. El conocimiento de las fases del desarrollo de un compilador tanto a nivel teórico como las dificultades a nivel práctico.

En cuanto a alumnos libres: se evaluará la capacidad del alumno para desarrollar un compilador académico siguiendo las pautas brindadas antes del examen y la capacidad del mismo en realizarle una modificación durante el examen. Se valorará la precisión del alumno en las respuestas a las preguntas realizadas durante el examen oral. La coherencia de sus respuestas entre sí junto con el grado de conocimiento demostrado en la capacidad de relacionar los conceptos de la asignatura. La madurez conceptual evidenciada en la capacidad de respuestas rápidas concisas y precisas a consultas realizadas. El conocimiento de las fases del desarrollo de un compilador tanto a nivel teórico como las dificultades a nivel práctico.

Condiciones para la acreditación de la asignatura:



Examen Final alumnos regulares. El examen final es de tipo integrador teórico-práctico, de forma oral o escrita, sobre cualquiera de los temas desarrollados en la materia. El examen final se rinde a programa completo, exigiéndose el nivel superior correspondiente, independientemente que se hayan tomado o no en las evaluaciones parciales. Todos los temas evaluados deben conocerse en al menos un 60% del alcance desarrollado en la materia. Podrán rendir examen final aquellos alumnos regulares. La calificación del examen final considerará la totalidad del proceso de enseñanza aprendizaje.

Examen final para alumnos libres. El alumno deberá presentarse con su compilador siguiendo las consignas dadas durante el año lectivo en curso con el agregado de alguna funcionalidad al mismo solicitado por el docente responsable antes de presentarse al examen final. El examen consta de la evaluación del compilador y un examen oral. El compilador deberá ser entregado al profesor responsable de la asignatura 72 hs. antes del examen final. En caso de que el compilador esté aprobado con una nota superior o igual a 6 (SEIS) el alumno libre deberá rendir un examen oral en el mismo momento y lugar según los criterios de evaluación mencionados previamente. El examen final libre se rinde a programa completo, exigiéndose el nivel superior correspondiente.

Programa de examen:

Contempla la totalidad de los temas del presente programa.

FECHA, FIRMA Y ACLARACIÓN RESPONSABLE DE CÁTEDRA