

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO UNIVERSITÁRIO NORTE DO ESPÍRITO SANTO

PROGRAMA DE DISCIPLINA

DEPARTAMENTO DE ENGENHARIAS E COMPUTAÇÃO -DECOM					
DISCIPLINA OU ESTÁGIO: Compiladores					CÓDIGO: DEC08118
CRÉDITOS	CH	T	E	L	PRÉ/CO/REQUISITOS
04	60	60	-	-	Linguagens de Programação, Arquitetura de Computadores, Linguagens Formais e Autômatos

EMENTA (Tópicos que caracterizam as unidades dos programas de ensino)

Organização e estrutura de compiladores e interpretadores. Análise léxica. Análise sintática. Recuperação de erros. Tabela de símbolos. Análise semântica. Geração de código intermediário. Otimização de código. Geração de código destino. Construção de um compilador completo apoiado por ferramentas do tipo compiler-to-compiler..

OBJETIVOS GERAIS

- Investigar os fundamentos da construção de compiladores.
- Descrever a estrutura e o funcionamento de compiladores.
- Desenvolver implementação de todas as fases da compilação.
- Desenvolver competências necessárias para aplicar as técnicas de compilação na resolução de problemas em diversas áreas.

CONTEÚDO PROGRAMÁTICO

1. Tradutor, interpretador, compilador: definições, características e aplicações; processo de compilação; estrutura geral e funcional de um compilador.

2. Analisador léxico: lexemas; tokens; análise linear; o analisador léxico como um autômato finito; projeto de um analisador léxico; expressões regulares para definição do tokens; tratamento de erros léxicos.

3. Analisador sintático: análise hierárquica; gramática livre de contexto para análise sintática, gramática EBNF; precedência de operadores; compilação dirigida pela análise sintática; tratamento de erros sintáticos.

4. Análise Sintática Top-down: análise descendente top-down; analisador recursivo com retrocesso; analisador recursivo preditivo; analisador tabular preditivo.

5. Análise Sintática Bottom-up: análise redutiva e ascendente; analisador redutivo ou empilha-reduz; analisadores LR.

6. Recuperação de Erros: método do pânico; método de recuperação local; produção de erro reconhecido pela gramática.

7. Tabela de símbolos: tabela hashing como estrutura de dados adequada; atributos da tabela, operações com a tabela através de ações semânticas na gramática.

8. Analisador semântico: ações semânticas associadas à gramática da linguagem; analisador semântico; analisador de contexto; checagem de tipos; tratamento de erros semânticos e de possíveis erros (warnings).

9. Gerador de código intermediário: estruturas de dados para a linguagem intermediária; ações semânticas para criação da linguagem intermediária; linguagem pós fixa para expressões.

10. Otimizador de código: otimização nas expressões com constantes numéricas e strings, operador de negação.

11. Persistência de código intermediário: armazenamento em memória secundária da estrutura que representa a linguagem intermediária.

12. Gerador de código destino: linguagem de baixo nível; linguagem de compiladores híbridos; instruções da JVM; construção de algoritmo conversor da linguagem intermediária para a linguagem destino.

13. Montagem e execução: montadores; uso do montador Jasmin; testes com a JRE; depuração.

14. Gerador automático de analisador léxico e sintático: compiler-to-compiler para tipos diferentes de análises sintáticas; uso do Javacc ao longo do processo de construção de um compilador híbrido completo.

BIBLIOGRAFIA BÁSICA

AHO, Alfred V.; LAM, Monica S.; SETHI, Ravi; ULLMAN, Jeffrey D. **Compiladores: princípios, técnicas e ferramentas.** 2 ed. São Paulo: Pearson, 2008.

PRICE, Ana Maria de Alencar; TOSCANI, Simão Sirineo. **Implementação de linguagens de programação: compiladores.** 3. ed. Porto Alegre: Sagra Luzzatto, 2008.

LOUDEN, Kenneth C. **Compiladores: princípios e práticas.** São Paulo: Thomson, 2004.

BIBLIOGRAFIA COMPLEMENTAR

DELAMARO, Márcio Eduardo. **Como construir um compilador: utilizando ferramentas Java**. São Paulo: Novatec, 2004.

GRUNE, Dick et al. **Projeto moderno de compiladores: implementação e aplicações**. Rio de Janeiro: Campus, 2001.

MORGAN, Rober. **Building on Optimizing Compiler**. Boston: Butterworth-Heinemann, 1998.

APPEL, Andrew; GINSHURG, Maia. **Modern compiler Implementation**. C. Austrália: Cambridge University Press, 1988.

SETZER, V. W; MELO I. S. H. **A construção de um compilador**. 3. ed. Rio de Janeiro: Campus, 1983.

VAREJÃO, F. M. **Linguagens de programação: conceitos e técnicas**. Rio de Janeiro: Campus, 2004.

JAVACC. Documentation. Disponível em <<http://javacc.java.net/doc/docindex.html>>

NORVELL, Theodore S. The JavaCC FAQ. Computer and Electrical Engineering. Memorial University of Newfoundland. 2011. Disponível em <<http://www.engr.mun.ca/~theo/JavaCC-FAQ/javacc-faq-moz.htm>>

ORACLE. The Java language specification and the Java virtual machine specification. Disponível em <<http://docs.oracle.com/javase/specs/index.html>>

METODOLOGIAS (aula expositiva, seminários, pesquisa, etc)

Aula expositiva e dialógica. Resolução de exercícios para apoio à aprendizagem. Construção cooperativa de um compilador completo: planejamento e implementação baseada no Modelo de Desenvolvimento Ágil através de técnica ASD (Adaptative Software Development) da Engenharia de Software. Prontuário do professor: anotações individuais e em grupo, divisão de tarefas e responsabilidades para a construção cooperativa. Uso de ambiente virtual cooperativo para apoio ao desenvolvimento e controle de versões. Estudo, pesquisa e elaboração de seminários sobre parte do conteúdo teórico com auxílio de mapas conceituais para apoio à aprendizagem significativa.

RECURSOS (audiovisual, periódicos, material laboratório, etc)

Ambiente para controle de versões. Uso do AVA (ambiente virtual de aprendizagem) para interações de apoio à aprendizagem entre professor-aluno-aluno via fórum assíncrono, compartilhamento de notas de aula, exercícios e roteiro para a construção do compilador. Bibliografia disponível na biblioteca do campus. Projetor. Computadores. Softwares diversos para construção do compilador e representação de conhecimento por meio de mapas conceituais.

CRITÉRIOS DE AVALIAÇÃO DA APRENDIZAGEM

- 1ª avaliação (3,0 pontos): participações na construção cooperativa do compilador.
- 2ª avaliação (1,5 pontos): prova individual sobre a construção do compilador.
- 3ª avaliação (1,0 ponto): seminário em grupo.
- 4ª avaliação (1,5 pontos): prova individual sobre os assuntos do seminário.
- 5ª avaliação (3,0 pontos): prova prática e individual sobre todo o trabalho desenvolvido.

Observações quanto à 1ª avaliação: as participações poderão acontecer de forma individual ou em grupo e serão indicadas previamente pelo professor de acordo com as necessidades momentâneas da construção cooperativa do compilador. A avaliação da participação ocorrerá através da apresentação, para a turma, dos elementos indicados e implementados, que deverão ser disponibilizados em seguida no ambiente de controle de versões para a continuidade cooperativa do trabalho.

A média semestral será a soma dos pontos obtidos nessas cinco avaliações. Somente será dispensado da prova final o aluno que obtiver média semestral maior ou igual a sete pontos.

<p>CÂMARA DEPARTAMENTAL DATA: <u>04 / 11 / 2016</u></p>	<p>COLEGIADO DE CURSO DATA: <u>08 / 11 / 16</u></p>	<p>ASSINATURA (S) DO(S) RESPONSÁVEL(EIS)</p>
<p><i>Leandro José Silvestre</i> Professor / SIAPE: 1504334 DOEL/CEUNES/UFES</p>	<p><i>[Assinatura]</i> Coordenador de Curso Bacharelado em Ciência da Computação CEUNES/UFES</p>	<p><i>[Assinatura]</i> Henrique Monteiro Cristóvão Professor / SIAPE: 1727365 DOEL/CEUNES/UFES</p>

Leandro José Silvestre
Professor / SIAPE: 1504334
DOEL/CEUNES/UFES
Sua tarefa do OCEL