MINISTÉRIO DA EDUCAÇÃO
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA CELSO SUCKOW DA FONSECA
DIRETORIA DE ENSINO (DIREN)
DEPARTAMENTO DE ENSINO SUPERIOR (DEPES)
DEPARTAMENTO DE INFORMÁTICA (DEPIN)
CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS PARA INTERNET (CST-SI)

DEPARTAMENTO/COORDENAÇÃO	PLANO DE CURSO DA DISCIPLINA
DEPIN - Departamento Acadêmico de Informática	**COMPILADORES**

CÓDIGO	PERÍODO	ANO	SEMESTRE	PRÉ-REQUISITOS
GTSI 1310	Opt	2012	2	GTSI1305 Teoria da Computação GTSI1222 Estruturas de Dados

CRÉDITOS	AULAS/SEMANA			TOTAL DE AULAS NO SEMESTRE
	TEÓRICA	PRÁTICA	ESTÁGIO	
4	4	0	0	72

EMENTA

Compiladores. Análise Léxica. Análise Sintática. Tradução Dirigida por sintaxe. Verificação de tipos. Geração de Código Intermediário. Otimização de Código. Geração de Código Objeto. Ferramentas para implementação de Compiladores.

BIBLIOGRAFIA

Bibliografia básica

1. AHO, A. V., LAM, M. S., SETHI, R. e ULLMAN, J. D. *Compiladores: princípios, técnicas e ferramentas.* Ed. Pearson.
2. PRICE, A. M. A. e TOSCANI, S. S. *Implementação de Linguagens de Programação: Compiladores*. Série Livros Didáticos UFRGS. Ed. Bookman.
3. LOUDEN, K. C. *Compiladores: princípios e práticas.* Ed. Thomson.

Bibliografia complementar

1. MENEZES, P. B. *Linguagens Formais e Autômatos.* Série Livros Didáticos UFRGS. Ed. Bookman.
2. GRUNE, D., BAL, H. E., JACOBS, C. J. H. e LANGENDOEN, K. G. *Projeto Moderno de Compiladores: implementação e aplicações.* Ed. Campus.
3. COOPER, K. D. e TORCZON, L. *Construindo Compiladores.* Ed. Campus/Elsevier.
4. SEBESTA, R. W. *Conceitos de Linguagens de Programação.* Ed. Bookman.
5. CORMEN, T. H., LEISERSON, C. E., RIVEST, R. L. e STEIN, C. Algoritmos – teoria e prática. Ed. Campus/Elsevier

OBJETIVO GERAL

Fornecer conhecimento sobre a teoria e prática para construção de compiladores, bem como conhecimento das técnicas de compilação em problemas diversos.

METODOLOGIA

- Aulas expositivas com recursos audiovisuais.

CRITÉRIO DE AVALIAÇÃO

A avaliação semestral envolve duas provas escritas (P1 e P2). As datas das provas são agendadas entre o professor e a turma. A média parcial (MP) será calculada pelo cômputo da média aritmética simples entre a nota P1 e P2:

$$MP = (P1 + P2) / 2$$

O aluno que faltar a uma das duas provas terá direito a uma avaliação alternativa, denominada segunda chamada, versando sobre todos os tópicos abordados no curso, e cuja data também é agendada entre docente e discentes. A nota obtida nessa 2ª chamada substituirá a da avaliação P1 ou P2 onde o aluno não esteve presente. Caso ele falte às duas avaliações, terá atribuído o grau ZERO em uma delas.

Segundo o regimento do CEFET-RJ, caso o aluno obtenha média parcial inferior a 3,0 (três e zero) estará reprovado diretamente. Graus MP maiores ou iguais a 7,0 (sete e zero) aprovam diretamente o aluno. Em situações onde o aluno tenha grau MP entre 3,0 inclusive e 7,0 exclusive, terá direito a uma prova final (PF), que, juntamente com a média parcial gerará uma nova média, denominada média final (MF). Essa média é calculada da seguinte forma:

$$MF = (MP + PF) / 2$$

Para ser aprovado, o aluno deve alcançar uma MF maior ou igual a 5,0 (cinco e zero). Caso contrário, estará reprovado, devendo repetir a componente curricular.

PROGRAMA

1. Compiladores
   - 1.1. Introdução
   - 1.2. Fases de um Compilador
   - 1.3. Análise do programa fonte
   - 1.4. Tabela de Símbolos
   - 1.5. Recuperação de Erros
   - 1.6. Síntese
2. Análise Léxica
   - 2.1. Objetivo do analisador léxico
   - 2.2. Gramáticas e Linguagens Regulares
   - 2.3. Especificação de *tokens*
   - 2.4. Especificação
   - 2.5. Implementação
   - 2.6. Tabela de Símbolos
3. Análise Sintática
   - 3.1. Objetivo do analisador sintático
   - 3.2. Gramáticas Livres-do-contexto
   - 3.3. Análise Descendente (*top-down*)

3.4. Análise Redutiva (*bottom-up*)

3.5. Analisadores de precedência de operadores

3.6. Analisadores LR

4. Tradução Dirigida por Sintaxe

4.1. Esquemas de tradução

4.2. Construção de Árvores Sintáticas

4.3. Esquemas de tradução

5. Verificação de Tipos
6. Geração de Código Intermediário
7. Otimização de Código
8. Geração de Código Objeto
9. Ferramentas para a implementação de compiladores