**PLANO DE ENSINO**

**CURSO:** Ciência da Computação

**SÉRIE:** 8º semestre

**DISCIPLINA:** Compiladores e Computabilidade – Oferecida em EaD

**CARGA HORÁRIA SEMANAL:** 1,5 horas-aula

**CARGA HORÁRIA SEMESTRAL:** 30 horas-aula

**I – EMENTA**

Conceitos básicos sobre compiladores e interpretadores. Tipos de compiladores. Análise léxica. Tabela de símbolos. Análise sintática. Tratamento de erros sintáticos. Análise semântica. Geração de código intermediário. Otimização. Geração de código objeto. Ambientes de execução.

**II – OBJETIVOS GERAIS**

A construção de compiladores é um ramo da ciência da computação cujo estudo proporciona conhecimentos importantes aos profissionais desta área, tais como, um exemplo de estruturação apropriada do problema, abordagem metodológica consistente necessária ao desenvolvimento de um projeto deste porte e experiências em sistemas que congregam diferentes conhecimentos específicos de computação.

Dadas as características de interpretação e tradução de informações, o estudo dos conceitos fundamentais envolvidos no projeto e construção de compiladores permite uma aplicação mais ampla das técnicas envolvidas, uma vez que torna o aluno familiarizado com os elementos necessários a escrita de interpretadores de comandos e programas de interface, bem como, para o processamento de dados estruturados e extensíveis.

**III – OBJETIVOS ESPECÍFICOS**

Familiarizar os alunos com as principais técnicas e conceitos envolvidos na compilação de programas de computador. Aprimorar seus conhecimentos sobre programação através do estudo da estrutura e características de uma linguagem de programação, bem como, das tarefas realizadas por um compilador para transformar os programas em seus equivalentes em linguagem de máquina.

Capacitá-lo no desenvolvimento de ferramentas de geração de programas, para avaliação e apoio aos processos de engenharia de software, interpretadores e processadores de dados estruturados, bem como, no desenvolvimento de novas linguagens e paradigmas de programação.

**IV – COMPETÊNCIAS**

Conhecer a organização e as operações básicas de um compilador. Compreender o impacto das características e paradigmas das diferentes linguagens de programação. Relacionar os fundamentos de programação com as tarefas desempenhadas pelo hardware, sendo capaz de compreender melhor a evolução das novas arquiteturas de computador.

**V – CONTEÚDO PROGRAMÁTICO**

1. Introdução à compilação
   1. As motivações para se estudar a construção de compiladores
   2. Descrição do processo realizado pelo computador a partir da codificação de um algoritmo pelo programador até a sua execução em um computador.
   3. O conceito de semântica e a importância da sua preservação.
   4. Caracterização das linguagens quanto ao nível de abstração.
2. Introdução à compilação - continuação
   1. Identificação e diferenciação dos elementos do domínio
      1. Linguagem fonte, de implementação e alvo
      2. Compiladores, tradutores, montadores e interpretadores
      3. IDEs, debuggers, editores gráficos de interface e demais ferramentas de apoio
3. Introdução à compilação - continuação
   1. Modelos para a construção de compiladores
      1. As etapas de transformação do código
      2. Compiladores de uma passagem e de múltiplas passagens
      3. O modelo de Análise e Síntese
      4. Discussão das vantagens e desvantagens de cada abordagem
4. Análise Léxica
   1. A transformação de fluxo de caracteres em lista de tokens
      1. A eliminação de elementos irrelevantes ao processo
      2. O emprego dos símbolos delimitadores na identificação e classificação dos tokens
   2. Autômatos finitos para reconhecimento e classificação de tokens
5. Análise Léxica - continuação
   1. Identificação de erros léxicos
      1. Destaque a quais tipos de erros pertencem e quais não pertencem a essa categoria
      2. Tratamento e recuperação de erros
   2. Tema para pesquisa: geradores de analisadores léxicos (LEX)
6. Análise Sintática
   1. Descrição da tarefa de verificação da estrutura sintática do programa
      1. Correlacionar o processo de análise sintática estabelecendo um paralelo com o processo de derivação de cadeias orientado pelas regras de produção de uma gramática
      2. Construção/simulação da árvore de derivação do programa
      3. O problema da utilização de gramáticas ambíguas
   2. Gramáticas livres de contexto
7. Análise Sintática - continuação
   1. Como é definida a precedência de operadores com arranjos de regras
   2. Técnicas de análise:
      1. Top-down: Descendente Recursiva e LL
      2. Bottom-up: SLR, LR e LALR
   3. Tratamento e recuperação de erros
   4. Tema para pesquisa: geradores de analisadores sintáticos (YACC)
8. Análise Semântica
   1. Gramáticas de Atributos
   2. Grafos de dependência e tratamento de ciclos
   3. Tabela de Símbolos
9. Ambientes de Execução
   1. Organização da memória
   2. Ambientes de execução
      1. Estáticos
      2. Pilha
      3. Dinâmicos
   3. Passagem de parâmetros
10. Geração de Código
    1. Geração de Código Intermediário
    2. Notações pré e pós-fixadas
    3. Código de três endereços
11. Geração de Código - continuação
    1. Blocos básicos e grafos de fluxo
    2. Otimização
    3. Geração em Linguagem Simbólica
12. Assemblers, linkeditores e carregadores
    1. Montadores
    2. Carregadores e ligadores

**VI - ESTRATÉGIAS DE TRABALHO**

A disciplina é ministrada por meio de tele aulas, metodologias ativas e diversificadas apoiadas no Plano de Ensino. O desenvolvimento dos conceitos e conteúdos ocorre com apoio de propostas de leituras das Unidades Instrucionais disponíveis, de livros e artigos científicos básicos e complementares, exercícios, discussões em fórum/chats, sugestões de filmes, vídeos e demais recursos audiovisuais. Com o objetivo de aprofundar e enriquecer o domínio dos conhecimentos e incentivar a pesquisa, o docente pode propor trabalhos individuais ou em grupo, palestras, atividades complementares e práticas em diferentes cenários, que permitam aos alunos assimilarem os conhecimentos essenciais para sua formação. Há acesso a monitores no caso do aluno necessitar de suporte para seus estudos.

**VII – AVALIAÇÃO**

O processo de avaliação ocorre no laboratório de informática, através de agendamento para a realização das provas, e consiste de duas notas bimestrais. No primeiro bimestre o ambiente AVA disponibilizará uma lista de exercícios online cuja correção gerará a nota NP1. No segundo bimestre o ambiente AVA gerará uma prova impressa cuja correção (através de cartão de respostas) gerará a nota NP2. As provas Substitutiva e Exame também serão impressas e geradas pelo ambiente AVA.

**VIII – BIBLIOGRAFIA**

**Básica**

AHO, A. V.; LAM, M. S.; SETHI, R.; ULLMANN, J. D. Compiladores: princípios, técnicas e ferramentas - 2ª ed. - São Paulo: Pearson Addison-Wesley, 2008.

<https://plataforma.bvirtual.com.br/Leitor/Publicacao/280/pdf/0>. Acesso em: 11 nov. 2022

LOUDEN, K. C.; Compiladores: princípios e práticas - 1ª ed. - São Paulo: Cengage Learning, 2004.

<https://integrada.minhabiblioteca.com.br/books/9788522128532>. Acesso em: 11 nov. 2022

TOSCANI, Simao Sirineo; PRICE, Ana Maria de Alencar - Implementação de Linguagens de Programação - Compiladores - 3ª Ed. - Vol. 9 - Ed. Artmed, 2009.

**Complementar**

RICARTE, I.; Introdução à Compilação - 1ª ed. - Rio de Janeiro: Elsevier, 2008.

GRUNE, D.; BAL, H. E.; JACOBS, C. J. H.; LANGENDOEN, K. G.; Projeto Moderno de Compiladores Implementação e Aplicações - Rio de Janeiro: Campus, 2001.

APPEL, A. W. - Modern compiler implementation in Java. - London: Cambridge University Press, 1998.

EPSTEIN, Richard L. CARNIELLI, Walter. Computabilidade, Funções Computáveis, Lógica e os Fundamentos da Matemática. Ed. [UNESP](http://www.livrariacultura.com.br/scripts/cultura/catalogo/busca.asp?parceiro=TIOIOO&tipo_pesq=editora&neditora=3526&refino=2&sid=01962248012929643876157446&k5=20672A2F&uid=), 2009.

NIES, Andre. Computability and Randomness. Ed. Oxford, UK, 2009.