DISCIPLINA:	COMPILADORES	
Código:	CPL	
Carga Horária Total: 80	CH Teórica: 60	CH Prática: 20
Número de Créditos:	4	
Pré-requisitos:	TEORIA DA COMPUTAÇÃO	
Semestre:	8º	
Nível:	Superior	

EMENTA

Introdução aos compiladores, análise léxica, análise sintática, geração do código intermediário.

OBJETIVO

Propiciar ao discente as ferramentas necessárias para a construção de compiladores, incluindo suas etapas de análise léxica, análise sintática, e geração do código intermediário.

PROGRAMA

Unidade I - Introdução aos compiladores:
- Evolução das linguagens de programação
- Tradutores e sua estrutura
- Análise léxica
- Análise sintática e semântica
- Geração do código intermediário
- Geração do código objeto
- Tabelas de símbolos
- Erros
- Geradores de compiladores

Unidade II - Análise Léxica:
- Tokens
- Especificação
- Implementação
- Tabela de símbolos

Unidade III - Análise Sintática:
- Análise descendente e descendente
- Recuperação de erros
- Implementação

Unidade IV - Geração do código intermediário:
- Linguagens intermediárias

- Ações semânticas
- Geração de código para comando de atribuição
- Expressões lógicas e comandos de controle
- Backpatching

METODOLOGIA DE ENSINO

Aulas teóricas:

- Ministradas em sala, ou outro ambiente que facilite o processo de ensino-aprendizagem, por meio expositivo-dialógico e com discussões com resolução de exercícios, onde a ênfase está em demonstrações conceituais e fundamentos essenciais;
- Como recursos de apoio, tem-se a utilização do quadro branco, projetor de slides e livro(s) de referência(s)

Aulas práticas:

- Ministradas em laboratório de informática, ou outro ambiente que facilite a consolidação dos conceitos fundamentais, por meio do uso e melhoramento de suas habilidades de trabalho ativo, onde a ênfase está na reflexão sobre o que se faz, provocando o encontro de significados no que for visto na aula teórica.
- Como recursos de apoio, tem-se a utilização de ferramentas de programação, de plataformas online de ensino aprendizagem de compiladores e trabalhos dirigidos à reprodução de aplicações rápidas para implementação de um pequeno compilador para uma linguagem de programação simples, ou parte dele, utilizando os conceitos da disciplina

Prática Profissional Supervisionada e projetos interdisciplinares:

- A PPS compreende diferentes situações de vivência profissional, aprendizagem e trabalho, por meio de experiências profissionais supervisionadas pelo professor, onde a ênfase é o estímulo à consolidação de um perfil pró-ativo, com a autoconfiança necessária para uma atuação profissional protagonista
- Deverá ser dada prioridade à realização de projetos interdisciplinares, tais como, por exemplo, em conjunto com a disciplina de teoria da computação e/ou linguagens de programação, possibilitando o diálogo entre diferentes disciplinas ou turmas, de maneira a integrar os conhecimentos distintos e com o objetivo de dar sentido a eles.
- Como sugestão de recursos de apoio, tem-se a realização de projetos finais para a disciplina, investigação sobre atividades profissionais, projetos de pesquisa ou outros trabalhos acadêmicos, visitas técnicas, simulações e observações as quais deverão ser desenvolvidas nos diversos ambientes de aprendizagem, como oficinas, incubadoras, empresas pedagógicas ou salas na própria instituição de ensino ou em entidade parceira

AVALIAÇÃO

O processo avaliativo deve ser contínuo e constante durante todo o processo de ensino-aprendizagem, com o propósito de analisar o progresso do aluno, criando indicadores

capazes de apontar meios para ajudá-lo na construção do conhecimento.
Desta forma, para início do processo ensino-aprendizagem, sugere-se avaliações diagnósticas, como forma de se construir um panorama sobre as necessidades dos alunos e, a partir disso, estabelecer estratégias pedagógicas adequadas e trabalhar para desenvolvê-los, inclusive evidenciando os casos que necessitarão de métodos diferenciados em razão de suas especificidades, tais como a necessidade de inclusão. Essas avaliações deverão seguir, preferencialmente, métodos qualitativos, todavia, também seguirão métodos quantitativos quando cabíveis dentro dos contextos individuais e coletivos da turma.
Durante toda a continuidade do processo ensino-aprendizagem, sugere-se a promoção, em alta frequência, de avaliações formativas capazes de proporcionar ao docente um feedback imediato de como estão as interferências pedagógicas em sala de aula, e permitindo ao aluno uma reflexão sobre ele mesmo, exigindo autoconhecimento e controle sobre a sua responsabilidade, frente aos conteúdos já vistos em aula, privilegiando a preocupação com a satisfação pessoal do aluno e juntando informações importantes para mudanças na metodologia e intervenções decisivas na construção de conhecimento dos discentes, inclusive com subsídios para propostas de atividades de recuperação paralela na(s) reunião(ões) de colegiado de curso, coordenadoria de curso e demais setores ligados ao ensino.
Ao final de cada etapa do período letivo, pode-se realizar avaliações somativas, com o objetivo de identificar o rendimento alcançado tendo como referência os objetivos previstos para a disciplina. Há nesses momentos a oportunidade de utilizar recursos quantitativos, tais como exames objetivos ou subjetivos, inclusive com recursos de TIC, todavia, recomenda-se a busca por métodos qualitativos, baseados no planejamento de projetos práticos, práticas interdisciplinares ou atuação em experimentos de laboratório, dentre outros.

BIBLIOGRAFIA BÁSICA

AHO, Alfred V. ... [et al.]. **Compiladores: princípios, técnicas e ferramentas - 2ª edição.** Pearson. E-book. (648 p.).

SANTOS, Pedro Reis. **Compiladores - Da Teoria à Prática.** LTC. (364 p.). ISBN: 9788521634829.

NETO, João José. **Introdução à compilação.** Elsevier Brasil, 2017.(328 p.). ISBN: 9788535278101

BIBLIOGRAFIA COMPLEMENTAR

BERGMANN, Seth D. **Compiler Design:** Theory, Tools and Examples. Open Educational Resources, 2017. Disponível em: https://rdw.rowan.edu/oer/1/. Acesso em 27 nov. 2021.

DONNELLY, Charles; STALLMAN, Richard. **Bison**: The Yacc-compatible Parser Generator. Boston, USA: Free Software Foundation, 2021. ISBN: 1-882114-44-2. Disponível em: http://www.gnu.org/software/bison/manual/bison.pdf. Acesso em: 27 nov. 2021.

AABY, Anthony A. **Compiler Construction using Flex and Bison**. Walla Walla College,

2003. Disponível em: https://www.admb-project.org/tools/flex/compiler.pdf. Acesso em: 27 nov. 2021.

GRANT, Mike; PALMER, Zachary; SMITH, Scott. **Principles of Programming Languages**. 2020. Disponível em: https://pl.cs.jhu.edu/pl/book/book.pdf. Acesso em: 27 nov. 2021.

MENEZES, Paulo Blauth. **Linguagens formais e autômatos.** Sagra-Dcluzzato, 1998.

KALINOVSKY, Alex. **Java Secreto: técnicas de descompilação, patching e engenharia reversa.** Pearson. E-book. (270 p.). ISBN 9788534615396.

Coordenador do Curso	**Setor Pedagógico**
_____	_____