

Campus: São José dos Campos		
Curso (s): Engenharia de Computação		
Unidade Curricular (UC): Laboratório de Sistemas Computacionais: Compiladores		
Unidade Curricular (UC): <i>Laboratory of Computer Systems: Compilers</i>		
Código da UC: 6098		
Docente Responsável/Departamento: Luiz Eduardo Galvão Martins / Departamento de Ciência e Tecnologia		Contato (e-mail): <i>legmartins@unifesp.br</i>
Docente (s) Colaborador/a (es/as)/Departamento (s):		Contato (e-mail): [opcional]
Ano letivo: 2024	Termo: 7º	Turno/Turma: Integral
Nome do Grupo/Módulo/Eixo da UC (se houver):		Idioma predominante em que a UC será oferecida: <input checked="" type="checkbox"/> Português <input type="checkbox"/> English <input type="checkbox"/> Español <input type="checkbox"/> Français <input type="checkbox"/> Libras <input type="checkbox"/> Outro:
UC: <input checked="" type="checkbox"/> Fixa <input type="checkbox"/> Eletiva <input type="checkbox"/> Optativa	Oferecida como: <input checked="" type="checkbox"/> Disciplina <input type="checkbox"/> Módulo <input type="checkbox"/> Estágio <input type="checkbox"/> Outro:	Oferta da UC: <input checked="" type="checkbox"/> Semestral <input type="checkbox"/> Anual
Ambiente Virtual de Aprendizagem: <input checked="" type="checkbox"/> Moodle <input type="checkbox"/> Classroom <input type="checkbox"/> Outro: <input type="checkbox"/> Não se aplica		
Pré-Requisito (s) - Indicar Código e Nome (s) da (s) UC: 2615 (Compiladores); 6095 (Laboratório de Sistemas Computacionais: Engenharia de Sistemas)		
Carga horária total (em horas): 72		
Carga horária teórica (em horas): 14	Carga horária prática (em horas): 58	Carga horária de extensão (em horas, se houver): não há
Se houver atividades de extensão, indicar código e nome do projeto ou programa vinculado na Pró-Reitoria de Extensão e Cultura (ProEC): Não há.		
Ementa: Ambientes de execução. Conjunto de instruções (nível ISA). O processo de síntese do compilador. Geração de código objeto. Otimização de código		
Conteúdo programático: Organização de memória durante a execução de programas. Ambientes de execução estáticos. Ambientes de execução baseados em pilhas. Memória dinâmica. Mecanismos de passagem de parâmetros. Código intermediário e estruturas de dados para geração de código. Técnicas básicas para geração de código. A linguagem objeto. Endereços no código objeto. Alocação e atribuição de registradores. Técnicas de otimização de código. Otimizações independentes de máquina.		
Objetivos:  <u> Gerais:</u> O objetivo geral dessa unidade curricular é capacitar o aluno a construir um compilador completo, envolvendo o processo de análise e síntese do compilador.  <u> Específicos:</u> Capacitar o aluno a especificar a linguagem de programação de alto nível, para a qual o compilador será construído; Capacitar o aluno na especificação e modelagem do compilador a ser implementado; Construir os módulos de análise léxica, sintática e semântica do compilador; Construir os módulos de geração e otimização de código objeto da máquina alvo; Capacitar o aluno a desenvolver apresentações orais e redação de textos relativos aos conteúdos trabalhados na unidade curricular.		

#### Metodologia de ensino:

Esta unidade curricular será baseada na exposição dos conteúdos necessários para a realização da síntese do compilador e desenvolvimento de projeto. O projeto será realizado tanto em sala de aula como extraclasse, utilizando-se ferramentas de modelagem, compiladores e geradores automáticos de módulos de um compilador (léxico e sintático). Essa unidade curricular também levará o aluno a elaborar apresentações orais, construir estruturas de trabalhos técnicos e científicos, na forma de relatórios, além da redação de textos.

#### Avaliação:

O conceito final será computado com base na avaliação de dois entregáveis ao longo do semestre.

O primeiro entregável será composto por (valendo 3 pontos):

- Módulos de análise léxica, sintática e semântica do Compilador (código fonte);
- Módulo de geração de código intermediário do Compilador (código fonte);
- Modelos SysML correspondentes (diagramas de atividades e de blocos);
- Exemplos de geração de código intermediário, a partir de programas C-.

O segundo entregável será composto por (valendo 7 pontos):

- Relatório final (conforme *template* a ser disponibilizado);
- Código fonte completo do Compilador (envolvendo todos os módulos de análise de síntese).

Em ambos entregáveis, os alunos deverão fazer uma apresentação presencial para o professor.

#### Bibliografia:

##### Básica:

1. LOUDEN, Kenneth C; SILVA, Flávio S.c. Compiladores: princípios e práticas. São Paulo: Thomson, 2004. 569 p. ISBN 9788522104222.
2. AHO, Alfred V; ULLMAN, Jeffrey D; SETHI, Ravi; LAM, Monica S. Compiladores: princípios, técnicas e ferramentas. 2 ed. São Paulo: Person Addison Wesley, 2007. 634 p. ISBN 9788588639249.
3. APPEL, Andrew W; PALSBERG, Jens. Modern compiler implementation in Java. 2nd ed. New York: Cambridge University Press, 2002. 501 p ISBN 9780521820608.

##### Complementar:

1. SCOTT, Michael L. Programming language pragmatics. New York: Morgan Kaufmann, c2009. 910 p. ISBN 9780123745149.
2. SANTOS, Pedro Reis. Compiladores: da teoria à prática. Rio de Janeiro LTC 2018 1 recurso online ISBN 9788521635161.
3. ULLMAN, Jeffrey D; MOTWANI, Rajeev; HOPCROFT, John E. Introduction to automata theory, languages, and computation. 3rd ed. Boston: Pearson, 2006. 535 p. ISBN 9780321455369.
4. RICARTE, Ivan. Introdução à compilação. Rio de Janeiro: Elsevier, 2008. 264 p. ISBN 9788535230673.
5. PRICE, Ana Maria de Alencar; TOSCANI, Simão Sirineo. Implementação de linguagens de programação: compiladores. 3.ed. Porto Alegre: Bookman, 2008. 195 p. ISBN 9788577803484.
6. WAZLAWICK, Raul Sidnei. Metodologia de pesquisa para ciência da computação. 3. Rio de Janeiro GEN LTC 2020 1 recurso online ISBN 9788595157712.