Plano de Atividades Domiciliares ADE

Unidade Curricular: Laboratório de Sistemas Computacionais: Compiladores

Professor(a): Thaína Aparecida Azevedo Tosta		**Contato**: tosta.thaina@unifesp.br
Ano letivo: 2021	**Semestre**: 1º	**Carga horária total**: 72hs (ADE = 72hs)
Turma: Noturno		

Plataformas de acesso ao curso:

Moodle/Google Classroom: repositório de atividades e material didático;
Google Meet: encontros síncronos semanais e plantões (sob demanda).

Objetivos:
Esta unidade curricular faz parte das unidades curriculares integradas definidas no Projeto Pedagógico do Curso, as quais são utilizadas para que o aluno possa, de fato, desenvolver um sistema computacional completo durante o seu processo de aprendizagem, envolvendo a integração entre hardware e software. O sistema completo compreende o desenvolvimento da arquitetura do processador, a definição de uma linguagem de programação, o projeto de um compilador, a definição de um sistema operacional e um processo de comunicação em rede entre dois ou mais sistemas. Dentro deste contexto, ao término desta unidade curricular, o aluno deverá ter implementado um compilador completo para o sistema computacional especificado. O objetivo geral dessa disciplina é capacitar o aluno a construir um compilador completo, envolvendo o processo de análise e síntese do compilador.

Conteúdo Programático e Cronograma:

Conteúdo	Práticas Pedagógicas	Carga horária
1. Revisão dos módulos de análise léxica, sintática e semântica apresentados na disciplina de Compiladores. Modelagem dos módulos de análise do Compilador: diagramas de atividades e de blocos (SysML).	Aula síncrona: apresentação da disciplina.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3

2. Revisão da arquitetura e conjunto de instruções do processador implementado (FPGA). Estudo e análise de projetos de semestres anteriores.	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
3. Definição dos tipos de quádruplas do código intermediário. Modelagem do módulo de geração do código intermediário: diagrama de atividades e de blocos.	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
4. Implementação do módulo de geração de código intermediário.	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
5. Implementação do módulo de geração de código intermediário.	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
6. Implementação do módulo de geração de código intermediário.	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
7. Implementação e testes do módulo de geração de código intermediário.	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
	Atividades avaliativas: Entrega do módulo de geração de código intermediário, como parte do projeto da disciplina.	
8. Implementação dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
9. Implementação e testes dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
10. Implementação e testes dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
11. Implementação e testes dos módulos de síntese (gerador de código assembly e	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1

gerador de código binário).	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
12. Implementação e testes dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
13. Implementação e testes dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
14. Implementação e testes dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
15. Implementação e testes dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
	Atividades avaliativas: Elaboração do relatório do projeto da disciplina.	
16. Implementação e testes dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
	Atividades avaliativas: Elaboração do relatório do projeto da disciplina.	
17. Implementação e testes dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
	Atividades avaliativas: Elaboração do relatório do projeto da disciplina; Entrega final do projeto da disciplina.	
18. Implementação e testes dos módulos de síntese (gerador de código assembly e gerador de código binário).	Aula síncrona: orientação, acompanhamento, discussão e resolução de dúvidas em atendimento a discentes.	1
	Atividades assíncronas: desenvolvimento do conteúdo da semana.	3
	Atividades avaliativas: atividades de recuperação.	

Metodologia de Ensino Utilizada:
Aula síncrona: 1 hora por semana;
Atividades assíncronas (3 horas por semana):

- Modelagem e implementação do compilador;
- Projeto da disciplina.

Metodologia de Avaliação (estratégias para atingir conceitos "cumprido" ou "não cumprido"):

Avaliação do projeto da disciplina: nota igual ou superior a 6.

Bibliografia básica e complementar para uso remoto:

LOUDEN, Kenneth C. Compiladores: princípios e práticas. São Paulo Cengage Learning 2004 1 recurso online ISBN 9788522128532. (https://integrada.minhabiblioteca.com.br/#/books/9788522128532)

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 978-85-88639-24-9.

APPEL, Andrew W; PALSBERG, Jens. Modern compiler implementation in Java. 2 ed. New York: Cambridge University Press, 2002. 501 p ISBN 978-0-521-82060-8.

SCOTT, Michael L. Programming language pragmatics. New York: Morgan Kaufmann, c2009. 910 p. ISBN 978-0-12-374514-9.

HOPCROFT, John E; MOTWANI, Rajeev; ULLMAN, Jeffrey D. Introdução à teoria de autômatos, linguagens e computação. Rio de Janeiro: Campus, 2002. 560 p. ISBN 978-85-352-1072-9.

ULLMAN, Jeffrey D; MOTWANI, Rajeev; HOPCROFT, John E. Introduction to automata theory, languages, and computation. 3.ed. Boston: Pearson, 2006. 535 p. ISBN 978-0-321-45536-9.

PRICE, Ana Maria de Alencar; TOSCANI, Simão Sirineo. Implementação de linguagens de programação: compiladores. 3.ed. Porto Alegre: Bookman, 2008. 195 p. ISBN 978-85-7780-348-4.