CURSO:	Engenharia de Software		
DISCIPLINA:	Estruturas de Dados para Competições		
SEMESTRE/ANO:	2025/2		
CARGA HORÁRIA:	60 horas	**CRÉDITOS:**	04
PROFESSOR:	Edson Alves da Costa Júnior	**CÓDIGO:**	FCTE0003

PLANO DE ENSINO

1 Objetivos da Disciplina

A disciplina Estruturas de Dados para Competições tem como objetivo preparar os alunos do curso de Engenharia de Software da FCTE para competições de programação, como a Maratona de Programação. Estes eventos ampliam o horizonte dos alunos e os estimulam a se aprofundarem nos tópicos de programação em geral. Além disso, a disciplina também constitui mais uma oportunidade para estudo e aprimoramento dos alunos em programação, tornando-os engenheiros mais preparados e capazes de atuar com competência no mercado de trabalho.

2 Ementa do Programa

I. *Introdução*

i. Programação Competitiva
ii. Maratonas de Programação
iii. Juízes Eletrônicos
iv. Dicas para estudo e treinamento
v. Como começar

II. *Estrutura de Dados para Competições*

i. Estrutura de Dados Lineares
ii. Estrutura de Dados Não-Lineares
iii. Variantes
iv. Tópicos Avançados

3 Horário das aulas e atendimento

AULAS: terças e quintas, das 18:00 às 19:50 hrs.

ATENDIMENTO: segundas, das 12:30 às 14:30 hrs, via plataforma Teams.

4 Metodologia

A metodologia consiste em aulas expositivas, com o auxílio do quadro branco e projetor digital. A fim de fortalecer a aprendizagem da disciplina, as aulas serão complementadas com exercícios e atividades, presenciais e extra-classe. As comunicações do curso serão feitas exclusivamente através da plataforma SIGAA.

O curso também será focado na resolução de exercícios, envolvendo a análise e resolução de problemas oriundos de competições e de *online judges*. Ocasionalmente acontecerão contests ou na plataforma vJudge[1], ou na plataforma Codeforces[2], ou na plataforma AtCoder[3], ou na plataforma MOJ[4].

5 Critérios de Avaliação

A avaliação do curso se dará por meio de duas provas, individuais, cujas datas estão previstas no cronograma.

5.1 Provas

Cada prova será composta por 5 problemas. Por conta de potenciais problemas relacionados ao número de máquinas operantes no laboratório, a prova será aplicada em dois dias: a turma será dividida em dois grupos A e B, por meio de um sorteio, e o estudante fará a prova no dia reservado ao grupo no qual foi sorteado.

É permitida a consulta a materiais impressos e é vedada a consulta aos colegas ou a recursos online. A prova terá inicio às 18:20 hrs, e **não serão admitidos estudantes no ambiente de provas após às 18:30 hrs**.

A solução proposta para um problema será corrigida de acordo com os seguintes critérios: após ser compilada de forma bem sucedida, uma série de testes unitários automatizados alimentarão o programa resultante com entradas válidas e comparará os resultados obtidos com as saídas corretas. Uma solução será considerada aceita se obtiver sucesso em todos os testes unitários.

Após a aplicação da prova, as soluções propostas pelos estudantes serão avaliadas por ferramentas de identificação de plágio, e caso duas ou mais soluções apresentem índices de similaridade que caracterizem cópia, todas elas serão anuladas, mesmo que tenha recebido o veredito "Aceito" durante a prova.

A prova será realizada, a menos de dificuldades técnicas ou de indisponibilidade de equipamentos, nas máquinas do laboratório e em ambiente Linux, por meio do Nutella Boot do professor Bruno Ribas. As soluções para os problemas devem ser escritas em C, C++ ou Python. Soluções em outras linguagens não serão aceitas.

A menção final do curso será dada pelo total N de problemas cujas soluções foram aceitas, e não anuladas, nas duas provas, de acordo com a tabela abaixo.

[1] https://vjudge.net
[2] http://codeforces.com
[3] atcoder.jp
[4] https://moj.naquadah.com.br/cgi-bin/index.sh

N	**Menção**	**Descrição**
0	`SR`	*Sem rendimento*
1 ou 2	`II`	*Inferior*
3 ou 4	`MI`	*Médio inferior*
5 ou 6	`MM`	*Médio*
7 ou 8	`MS`	*Médio superior*
9 ou 10	`SS`	*Superior*

5.2 Listas de exercícios

A cada semana poderá será proposta uma lista de exercícios, com exercícios relacionados com o conteúdo ministrado. A resolução das listas não modifica a menção, mas é fortemente encorajada para a fixação dos conceitos apresentados no curso.

5.3 Atividades extras

O estudante poderá obter dois pontos extras a serem adicionados em nota N. O primeiro ponto poderá ser obtido por meio de participação presencial em um dos seguintes eventos de programação competitiva que acontecerão no segundo semestre de 2025:

1. Fase Subregional da Maratona de Programação da SBC
2. Fase Final da Maratona de Programação da SBC
3. XIII Maratona UnB de Programação
4. Maratona de Programação do IFB

O segundo ponto extra poderá ser obtido se o estudante atuar, como monitor, na aplicação de alguma das provas da disciplina. O aluno poderá se candidatar a monitor na aplicação da prova do grupo oposto ao que ele foi sorteado. Se houverem mais candidatos à monitor do que vagas, os monitores serão escolhidos mediante sorteio. Um estudante poderá ser monitor uma única vez.

5.4 Critérios de aprovação

Obterá **aprovação** no curso o aluno que cumprir as **duas** exigências abaixo:

1. Ter presença em 75% ou mais das aulas;
2. Obter menção igual ou superior a `MM`.

IMPORTANTE: Atestados médicos e documentos comprobatórios de justificativas de faltas dão direito à realização de atividades avaliativas que você venha a perder, mas essas ausências justificadas também são levadas em consideração como ausências efetivas para o cômputo da frequência mínima obrigatória (*Graduação UnB – Manual para estudantes*, pág. 35).

6 Cronograma

Semana	Aula	Data	Conteúdo
01	1	19/08	*Apresentação do curso*
	2	21/08	*Introdução à programação competitiva*
02	3	26/08	*Vetores*
	4	28/08	*Pilhas*
03	5	02/09	*Pilha monótona*
	6	04/09	*Fila*
04	7	09/09	*Fila monótona*
	8	11/09	*Árvores binárias*
05	9	16/09	*Conjuntos*
	10	18/09	*Venice Set*
06	11	23/09	*Dicionários*
	12	25/09	*Heaps*
07	-	30/09	**Prova 1A**
	-	02/10	**Prova 1B**
08	13	07/10	*Hashes*
	14	09/10	*Árvores de Fenwick: definição*
09	15	14/10	*Árvores de Fenwick: aplicações*
	16	16/10	*Árvores de Segmentos: definição*
10	17	21/10	*Árvores de Segmentos: aplicações*
	18	23/10	*Disjoint Sets Union*
11	19	28/10	*Sparse Table*
	20	30/10	*Link cut tree*
12	21	04/11	*Wavelet Tree*
	22	06/11	*Permutation Tree*
13	-	11/11	**Semana de Extensão Universitária**
	-	13/11	**Semana de Extensão Universitária**
14	23	18/11	*Sqrt Tree*
	-	20/11	**Feriado: Dia de Zumbi e Consciência Negra**

Semana	Aula	Data	Conteúdo
15	24	25/11	*Treap*
	25	27/11	*Interval Tree*
16	-	02/12	**Prova 2A**
	-	04/12	**Prova 2B**
17	-	09/12	**Prova Substitutiva**
	-	11/12	**Menções Finais**

7 Bibliografia

Livro Texto

HALIM, Steven S. and **HALIM**, Felix. *Competitive Programming*, 4ª ed, Lulu, 2010.

LAARKSONEN, A. *Competitive Programmer's Handbook*, Online, 2018.

ROUGHGARDEN, T. *Algorithms Illuminated (Part 3): Greedy Algorithms and Dynamic Programming*, Editora LLC, 2019.

Literatura Complementar

CORMEN, Thomas H. **LEISERSON** and Charles E. and **RIVEST**, Ronald L. and **STEIN**, Clifford. *Algoritmos: Teoria e Prática*, Editora Campus, 2ª ed, 2002.

DROZDEK, Adam. *Estruturas de Dados e Algoritmos em C++*, Thomsom, 2001.

KERNIGHAN, Brian and **RITCHIE**, Dennis M. *The C Programming Language*, Prentice Hall, 1988.

JOSUTTIS, Nicolai M. *The C++ Standard Library*, Addison-Wesley, 1999.

SOLTYS-KULINICZ, Michael. *Introduction to the Analysis of Algorithms*, World Scientific Publishing Co, 2012. (*eBrary*)

STEPHENS, Rod. *Essential Algorithms: A Practical Approach to Computer Algorithms*, John Wiley & Sons, 2013. (*eBrary*)

BALDWIN, Douglas; **SCRAGG,** Gregg. *Algorithms and Data Structures: The Science of Computing*, Charles River Media, 2004. (*eBrary*)