Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Informática e Estatística

Plano de Ensino

1) Identificação

Disciplina: INE5408 - Estruturas de Dados
Turma(s): 03208A
Carga horária: 108 horas-aula Teóricas: 54 Práticas: 54
Período: 2º semestre de 2019

2) Cursos

- Ciências da Computação (208)

3) Requisitos

- Ciências da Computação (208)
  - INE5404 - Programação Orientada a Objetos II

4) Professores

- Alexandre Goncalves Silva (alexandre.goncalves.silva@ufsc.br)
- Aldo Von Wangenheim (aldo.vw@ufsc.br)

5) Ementa

Alocação dinâmica de memória. Variáveis estáticas e dinâmicas. Estruturas lineares. Tabelas de Espalhamento. Árvores. Árvores de Pesquisa. Métodos de ordenação. Métodos de acesso a arquivos. Técnicas de implementações iterativas e recursivas de estruturas de dados. Complexidade dos algoritmos em estruturas de dados.

6) Objetivos

Geral: Capacitar o estudante a compreender, tanto do ponto de vista conceitual e teórico quanto prático (implementação e solução de problemas reais), as estruturas de dados clássicas a partir da perspectiva de diferentes paradigmas de programação.

Específicos:

- Identificar o papel das estruturas de dados no desenvolvimento de software.
- Compreender a teoria associada a cada modelo, sendo capaz de compreender e aplicar adequadamente aspectos de complexidade de algoritmos associados.
- Capacitar o aluno a criar uma biblioteca de estruturas de dados reutilizáveis.
- Identificar as estruturas de dados pertinentes a um problema dado.
- Transferir seus conhecimentos a problemas práticos do mundo real, sendo capaz de solucionar estes problemas práticos do mundo através da utilização das Estruturas de Dados adequadas, tanto do ponto de vista da eficácia quanto da eficiência.

7) Conteúdo Programático

7.1) Alocação dinâmica de memória [6 horas-aula]
- Variáveis estáticas e dinâmicas
- Ponteiros
- Passagem de parâmetros por referência, valor e nome

7.2) Estruturas lineares [24 horas-aula]
- Listas
- Pilhas
- Filas

7.3) Complexidade de algoritmos [6 horas-aula]
- Conceitos de Complexidade de Algoritmos
- Métodos práticos para Análise da Complexidade de Estruturas e Algoritmos

7.4) Árvores [24 horas-aula]
- Árvore binária

- Árvores binárias semibalancedas (Árvore AVL, Árvore Red-Black)
- Árvore semibalanceadas multivias (Árvore B e B+)
- Árvores binárias multichaves (Árvore k-d)
- Árvores semibalanceadas multivias multichaves (Árvores k-b-d e b-k-d)

7.5) Tabela de espalhamento (hash) [15 horas-aula]
- Tratamento de colisões
- Funções de espalhamento

7.6) Métodos de ordenação [12 horas-aula]
- Conceitos básicos, implicações e premissas
- Métodos de Complexidade Quadrática
  - Método por inserção
  - Método por seleção
  - Método da bolha
- Métodos Avançados e de Complexidade n log n
  - Método Quicksort
  - Método Heapsort

7.7) Estruturas de dados em arquivo [21 horas-aula]
- Acesso direto e Acesso sequencial
- Organização de Arquvos de Índices
  - Métodos indexados seqüenciais, ISAM e VSAM
  - Indexação por Árvore
  - Indexação por Multilistas e Lista Invertida

8) Metodologia

A disciplina terá um enfoque eminentemente prático. A metodologia de ensino será baseada no contraponto entre Aulas Teóricas e Aulas Práticas. Para tanto todo novo assunto será introduzido em uma aula teórica que terá a duração de 2 a 3 horas, acompanhada ou não pelo Estagiário de Docência designado para a disciplina. Este conteúdo teórico será fixado por meio de uma aula de caráter prático, com duração de 3 a 4 horas que contará com a proposição de um exercício que exige, além do horário em sala, mais de 3 a 4 horas extras classe para ser completado. As atividades práticas poderão ser executadas com auxílio de um Estagiário de Docência ou Monitor quando necessário, onde serão realizadas em laboratório atividades de modelagem e implementação com o objetivo de fixar o conteúdo, além da discussão em grupo de problemas de compreensão e implementação encontrados pela turma.

Como linguagem de programação para o ensino do conteúdo fica determinada a Linguagem C++ e como ambiente de implementação **obrigatoriamente** a Plataforma Linux/Unix, sendo a ferramenta de desenvolvimento preferencial o ambiente VPL do Moodle, podendo outras ferramentas serem utilizadas. Haverá uma breve introdução a ferramentas de produtividade nessas plataformas, à medida que tornarem necessárias, principalmente as de teste unitário, mas fica a cargo do aluno a responsabilidade de adquirir proficiência nas mesmas. O ensino dos conteúdos se dará por meio da utilização progressiva do Paradigma Orientado a Objetos, de acordo com a adequação à melhor compreensão do conteúdo.

A ferramenta de EAD Moodle disponível em moodle.ufsc.br será utilizada para guiar e organizar o ensino, sendo o repositório oficial de material de aula, devendo os alunos constantemente consultá-la para saber de novos exercícios e trabalhos requeridos. A disciplina no Moodle também detalhará o cronograma deste plano de ensino, servindo para marcar as datas exatas das avaliações e documentar alterações de cronograma advindas de necessidades identificadas no semestre. O fórum de discussão no ambiente do Moodle será utilizado para intermediar a comunicação entre professor, estagiário de docência, monitor e alunos. Para efeitos da avaliação da participação do aluno na disciplina, as suas estatísticas de utilização da ferramenta de EAD poderão ser levadas em consideração.

Como recurso didático auxiliar o professor poderá empregar vivências na forma de jogos de aprendizado (*serious games*), a serem jogados em equipe em sala de aula ou na forma de objetos de aprendizado digitais, disponibilizados via Moodle, para uso em casa ou na sala. A ferramenta Moodle ainda poderá ser utilizada para a entrega e avaliação automatizada, baseada em testes programáticos dos trabalhos de programação e de tarefa assinaladas pelo professor.

Pressupostos da metodologia de ensino

A metodologia adotada nesta disciplina pressupõe que os alunos do curso não se limitem a comparecer às aulas, mas que utilizem para as atividades extra-classe associadas a esta disciplina (leituras, resolução de exercícios teóricos e exercícios com o uso da ferramenta de EAD empregada) um número de horas igual ou superior ao dobro do número de horas-aula em sala de aula. Pressupõe-se que os alunos tenham estudado o conteúdo utilizando a bibliografia indicada e tenham resolvido, como atividade extra-classe, todos os exercícios propostos pelo professor.

Algumas aulas teóricas e atividades de laboratório serão acompanhadas pelo aluno de Pós-Graduação do

PPGCC, Jeferson Menegazzo, que estará realizando seu estágio de docência em 2019-2.

9) Avaliação

A avaliação regular dos alunos se dará por meio de um conjunto de 7 notas, sendo três destas notas referentes a trabalhos práticos, duas notas referente a provas teóricas e duas notas referentes a provas práticas de programação. A média final será composta pela média ponderada destas 7 notas conforme explanado abaixo:

Nota 1 (peso 2)- Calculada a partir da média simples das notas atribuídas a todos os pequenos trabalhos de fixação passados ao longo do semestre. Estes trabalhos estarão agendados no Moodle e serão em número entre 10 e 20, de acordo com a avaliação do professor do perfil de aprendizado da turma e suas dificuldades.
Nota 2 (peso 1) - Um Projeto de Implementação, denominado Projeto I, envolvendo um problema prático do mundo real solúvel com a aplicação de pilhas, listas ou filas e ponteiros e alocação dinâmica de memória. Este trabalho poderá ser realizado em equipe e poderá ser defendido pela equipe em casos onde não fique claro a participação de todos os membros da equipe, sendo atribuída nota individual a cada aluno.

Nota 3 (peso 1) – Uma prova de prática de programação individual denominada Prova Prática I, cobrindo os assuntos de estruturas lineares. A prova prática será feita utilizando a ferramenta Moodle na qual uma estrutura de dados deverá ser implementada e deverá passar por testes unitários automatizados de acordo com o preceitos utilizados em competições de olimpíadas de programação. A prova pode incluir restrições de tempo para implementação, tempo para execução e quantidade e técnicas de gerência de memória para execução.
Nota 4 (peso 1) - Um prova teórica individual denominada Prova Teórica I, cobrindo os assuntos de estruturas lineares e complexidade de algortimos, realizado em laboratório, eventualmente por meio do uso da ferramenta Moodle Provas, onde serão avaliados aspectos do conhecimento teórico.
Nota 5 (peso 2)- Um Projeto de Implementação, denominado Projeto II, envolvendo um problema prático do mundo real solúvel com a aplicação de Técnicas de Gerência de Arquivos Indexados. Este trabalho poderá ser realizado em equipe e poderá ser defendido pela equipe em casos onde não fique claro a participação de todos os membros da equipe, sendo atribuída nota individual a cada aluno.
Nota 6 (peso 1) – Uma prova de prática de programação individual denominada Prova Prática II. A prova prática será feita utilizando a ferramenta Moodle na qual uma estrutura de dados deverá ser implementada e deverá passar por testes unitários automatizados de acordo com o preceitos utilizados em competições de olimpíadas de programação. A prova pode incluir restrições de tempo para implementação, tempo para execução e quantidade e técnicas de gerência de memória para execução.
Nota 7 (peso 2) - Um prova teórica individual denominada prova teórica II, cobrindo todo o conteúdo da disciplina, realizado em laboratório, eventualmente por meio do uso da ferramenta Moodle Provas, onde serão avaliados aspectos do conhecimento teórico.

Todos os trabalhos deverão ser entregues juntamente com a documentação exigida pelo professor, sendo a avaliação realizada sobre código e documentação. A forma de entrega é até a data determinada e por meio do Moodle de acordo com os critérios estabelecidos para cada trabalho, podendo ser em arquivos compactados ou por meio da ferramenta de avaliação automática VPL, dependendo da indicação no enunciado do trabalho. É de responsabilidade do aluno entregar o trabalho na forma correta, arquivos corrompidos ou ilegíveis não serão considerados. Trabalhos entregues após o prazo sofrerão penalizações de 50% da nota independente do atraso.

A critério do Professor, sendo o desempenho parcial dos alunos considerado adequado e suficiente, o que será mensurado pela qualidade da maioria dos trabalhos entregues (condição mínima: média da turma >= 7.0), o item de avaliação número quatro poderá ser considerado desnecessário e omitido. Também a critério do professor, sendo o desempenho parcial dos alunos considerado adequado e suficiente, o que será mensurado pela qualidade da maioria dos trabalhos entregues (condição mínima: média da turma >= 7.0), o item de avaliação número sete poderá ser considerado desnecessário e omitido. Em função do caráter prático da disciplina, os itens 1, 2, 3, 5 e 6 sempre serão avaliados. Nesse caso, a média final transferirá o peso das provas teóricas para as provas práticas correspondentes.

Se o desempenho médio dos alunos durante o semestre for considerado extraordinário (condições mínimas: média geral da turma >= 7.0 ou média da turma no item de avaliação número três >= 7.0)), o professor pode optar por não exigir a defesa dos Projetos de Implementação, considerando, daquelas equipes que demonstraram bom aproveitamento em todos os aspectos, apenas código e documentação entregues para fins de avaliação.

Durante a defesa dos projetos de implementação o professor se reserva o direito de questionar individualmente os alunos da equipe sobre aspectos teóricos da disciplina contemplados no trabalho, sendo o resultado desses questionamentos levado em consideração de forma individual na atribuição do conceito.

Para avaliação dos trabalhos práticos o professor poderá trabalhar com dois modelos de correção: um manual e um automático, sendo dada preferência a este último sempre que possível.

Para os casos de de correção automática dos trabalhos de programação, estes serão avaliados por meio da

plataforma Moodle. Os testes estarão disponíveis no momento da entrega do trabalho pelos alunos. Os testes serão aplicados e uma nota final do trabalho será sugerida pela ferramenta de correção, podendo o aluno entregar o trabalho quantas vezes forem necessárias para a obtenção da nota desejada, tendo como limite a data de entrega final do trabalho em questão. Cabe ao professor a confirmação da notas em avaliação posterior. Neste caso a nota pode ser alterada levando em consideração a técnica, o estilo e a eficiência do código à critério do professor e previamente estabelecido no enunciado do trabalho.

Os trabalhos corrigidos automaticamente devem passar na sua integralidade e sem quaisquer defeitos nos testes programados para aquela entrega. Para tal os alunos devem sempre testar seus trabalho na plataforma de avaliação, não sendo aceitas reclamações pela não execução no ambiente disponibilizado. Os testes aplicados no processo de avaliação automática podem incluir:

Testes baseados na interface de usuários, onde dados são alimentados para o programa desenvolvido por meio da sua interface e o retorno é avaliado. Nestes casos a precisa implementação da interface é parte da avaliação.
Testes unitários em classes, onde os alunos serão instruídos a implementar uma interface de classe para que as classes entregues possam ser testadas método a método quanto ao seu funcionamento.
Avaliação estática de código buscando boas práticas de nomeação de atributos, de visibilidade dos métodos, de estruturação e documentação do código.
Avaliação dinâmica de código buscando boas práticas de gerência de memória e consistência no uso de métodos e classes em tempo de execução.
Avaliação de desempenho utilizando restrições de tempo e memória para a execução dos trabalhos no sistema de avaliação disponibilizado pela plataforma Moodle.

Os pesos de cada item de avaliação nos trabalhos corrigidos automaticamente será divulgado junto com o enunciado de cada trabalho e podem variar de acordo com o objetivo didático de cada trabalho.

No caso de correções manuais de trabalhos que não envolvam a defesa, serão utilizados 5 critérios objetivos, sendo que cada critério vale 2 pontos:

Compreensão do Problema: entendeu o que era para fazer (1 ponto) e modelou corretamente a solução (1 ponto) ?
Emprego dos Algoritmos e Estruturas: empregou os algoritmos e estruturas corretos (% dos algoritmos/estruturas necessários * 2 pontos) ?
Implementação dos Algoritmos e Estruturas: os implementou corretamente (% dos algoritmos e estruturas necessários * 2 pontos) ?
Código: compilou (1 ponto) e funcionou (1 ponto) ?
Documentação: código está documentado e programado como indicado no início do semestre (comentários 1 ponto, estilo de codificação 1 ponto)?

Havendo defesa dos Projetos de Implementação, a mesma é considerada Prova e o não comparecimento injustificado implica em conceito nulo nestes trabalhos, sendo os critérios de avaliação da defesa os abaixo:

Compreensão do Problema: entendeu o que era para fazer ? (2 pontos)
Solução: soube encontrar uma solução ? (2 pontos)
Conhecimento teórico: compreendeu as implicações teóricas da solução escolhida ? (2 pontos)
Algoritmos: possui compreensão dos algoritmos empregados e sabe descrevê-los ? (2 pontos)
Código: compreende a implementação, sabendo detalhar aspectos de seu funcionamento ou de falhas que estão ocorrendo ? (2 pontos)

Para efeitos de defesa, os trabalhos práticos dos alunos deverão estar disponíveis nos links de entrega disponibilizados no Moodle da disciplina, não podendo ser trocados ou atualizados quaisquer arquivos. Todas as defesas poderão ser realizadas na presença de uma Testemunha e as perguntas e suas respostas protocoladas por esta testemunha. Esta testemunha será preferencialmente o estagiário de docência mas poderá ser algum aluno de pós-graduação ou professor do INE.

Plágio e Deslealdade Acadêmica

Todos as atividades da disciplina seguirão um código ético severo, onde os trabalhos serão avaliados quanto a plágio de forma automatizada por uma ferramento que o professor dispõe. Caso esta ferramenta indique a possibilidade de plágio, o professor conferirá automaticamente a nota ZERO a todos os envolvidos e reportará ao órgão competente da universidade para a abertura de processo disciplinar e a aplicação das punições regimentais previstas.

Quanto aos casos de deslealdade acadêmica em provas (vulgarmente conhecido como colar), o professor lavrará um termo que será entregue ao órgão competente da universidade para a abertura de processo disciplinar e a aplicação das punições regimentais previstas.

Frequência na Disciplina

Para a avaliação da frequência na Disciplina fica rigidamente expresso o entendimento disposto na resolução 017/Cun/97 que determina a necessidade de 75% de presença para os cursos presenciais da Universidade.

Todos os registros de frequência serão feitos pela ferramenta Moodle e estarão disponíveis para consulta por parte do alunos.

Dado que a disciplina apresenta pelo menos 50% da carga horária consistindo de aulas práticas, conforme deliberação do Colegiado do Curso de Ciências da Computação de 18 de março de 2008, ela não prevê a realização de avaliação no final do semestre (recuperação) de que trata o parágrafo 2º do artigo 70 da Resolução 17/CUn/97.

10) Cronograma

Aula 1 - Plano de Ensino
Aula 2 - Introdução à Programação C++
Aula 3 - Pilha e Fila com Vetores(arrays)
Aula 4 - Programação orientada a objetos usando C++
Aula 5 - Lista utilizando vetores (arrays)
Aula 6 - Depuração de programas em ambientes de programação, criação de testes unitários, gerência e alocação dinâmica de memória
Aula 7 - Lista Encadeada
Aula 8 - Auxílio à Implementação
Aula 9 - Fila Encadeada e Pilha Encadeada
Aula 10 - Discussões sobre o trabalho de implementação I
Aula 11 - Lista Circular
Aula 12 - Auxílio na elaboração do Projeto de Implementação I
Prova Teórica I
Apresentação Trabalho I
Aula 13 - Conceitos de Complexidade de Algoritmos
Prova Prática I
Aula 14 - Árvores Binárias de Busca
Aula 15 - Auxilio com trabalhos pendentes
Aula 16 - Árvore AVL
Aula 17 - Auxilio com trabalhos pendentes
Aula 18 - Árvores Red-Black
Aula 19 - Árvores de Busca Semibalanceadas Multivias
Aula 20 - Hashing
Aula 21 - Gerência de Arquivos
Aula 22 - Gerencia de Arquivos - Índices Avançados
Aula 23 - Quicksort
Aula 24 - Discussões Trabalho de Implementação II
Aula 25 - Heapsort
Prova Teórica II
Apresentação Trabalho de Implementação II
Prova Prática II

11) Bibliografia Básica

- PEREIRA, Silvio do Lago. Estruturas de dados fundamentais: conceitos e aplicações. 12. ed., rev. e atual. São Paulo: Érica, 2009. 238 p. ISBN 9788571943704
- JOYANES AGUILAR, Luis. Programação em C++: algoritmos, estruturas de dados e objetos. São Paulo: McGraw Hill, 2008. xxxi, 768 p. ISBN 9788586804816
- WIRTH, Niklaus; ZÜRICH, Eth. Algoritmos e estruturas de dados. Rio de Janeiro: LTC, 1999. 255 p. ISBN 978-85-216-1190-5
- HOROWITZ, Ellis. Fundamentos de estruturas de dados.. Rio de Janeiro: Ed. Campus, 1986.
- CLAYBROOK, Billy G. (Billy Gene). Tecnicas de gerenciamento de arquivos. 2. ed. Rio de Janeiro: Campus, 1987. 248p. ISBN 8570012608

12) Bibliografia Complementar

- SILBERSCHATZ, KORTH E SUDARSHAN. Database System Concepts. 4ª ed., McGraw-Hill, 2001.
- THARP, A. File Organization and Processing. John Wiley, 1988.
- SEDGEWICK, R. Algorithms in C++. Addison Wesley, 1996.
- GOODRICH, M.T., TAMASSIA, R. Estruturas de Dados e Algoritmos em Java. 4a. Edição. Ed. Bookman, 2007.
- AMERAAL, L. Algorithms and Data Structures in C++. Editora John Wiley, 1996.
- TENNENBAUM, A., LANGSAM, Y. Data Structures using C and C++. 2ª Ed. Prentice-Hall, 1995.