ALGORITMOS AVANÇADOS - CCT0837

ALGORITMOS AVANÇADOS (09/10/2015)

Perfil Docente

Titulação:

Graduação em Ciência da Computação, Sistemas de Informação, Engenharia de Computação ou Licenciatura em Computação.

- Pós Graduação Lato Sensu.
- Currículo atualizado na Plataforma Lattes.

Desejável:

- Pós Graduação Stricto Sensu (Mestrado ou Doutorado) na área de computação.
- Experiência de três anos em docência de nível superior na disciplina.
- Experiência profissional de 5 anos no mercado de trabalho em Desenvolvimento de Software/Sistemas.

Perfil docente:

Além da titulação mínima necessária é importante que o docente tenha também a habilidade de articular os conteúdos vistos nas disciplinas de Introdução a Programação e Estrutura de Dados com os conteúdos que serão apresentados durante o decorrer da disciplina e sempre deixando ganchos daquilo que será ministrado nas Disciplinas que envolvem Programação de Computadores, mostrando ao aluno que a disciplina de Algoritmos Avançados não é uma disciplina isolada, mas faz parte de um processo de construção do saber do Profissional da Área de Desenvolvimento de Sistemas.

Contextualização

O estudo de Algoritmos Avançados, componente fundamental no aprendizado da Área da Computação, é a base sobre a qual muitos outros campos dessa área são construídos. O conhecimento dos Algoritmos Avançados é importante para estudantes que desejem trabalhar em implementações, testes e manutenção de projetos de qualquer sistema de software.

Ementa

Complexidade (notação O); recursividade; ordenações como mergesort e quicksort; noção de grafo, árvores binárias e AVL.

Objetivos Gerais

Desenvolver técnicas para representação de Algoritmos Avançados e as operações sobre os mesmos, de maneira que seja possível solucionar problemas, escolhendo os Algoritmos mais adequados para representação e manipulação dos dados em problemas específicos.

Objetivos Específicos

1. Identificar e conhecer a complexidade do elemento da análise assintótica ? Notação O.

2. Conhecer sobre recursividade e suas definições/implementações.
3. Analisar a complexidade do algoritmo de ordenação por intercalação (mergesort).
4. Analisar a complexidade do algoritmo de ordenação rápida (quicksort).
5. Conhecer o Algoritmo em Grafos e as suas representações.
6. Identificar e analisar as complexidades das estruturas de dados dos tipos Árvore Binária e Árvore AVL.

Conteúdos

UNIDADE I - Análise de Algoritmo - Notação O
1.1 - Algoritmo
1.2 - Estrutura de Dados
  1.2.1 - Revisão de Programas em C++ envolvendo Vetores, Matrizes, Ponteiros, Registros (Struct) e Funções.
1.3 - O que é Analise de Algoritmos
1.4 - Sobre o Elemento da Análise Assintótica - Notação O
  1.4.1 - Notação O
  1.4.2 - Sobre a função
  1.4.3 - Operações com a Notação O

UNIDADE II - Recursividade
2.1 - Definições recursivas
2.2 - Como implementar recursividade
2.3 - Quando não usar recursividade
2.4 - Desenvolvendo algoritmos com recursividade

UNIDADE III - Algoritmo de ordenação por intercalação (mergesort)
3.1 - Definição
3.2 - Dividir para conquistar
3.3 - Problema da intercalação
3.4 - O algoritmo de ordenação por intercalação mergesort.
3.5 - Análise da complexidade do algoritmo mergesort.

UNIDADE IV - Algoritmo de ordenação rápida (quicksort)
3.1 - Definição
3.2 - Ordenação rápida
3.3 - O algoritmo de ordenação rápida quicksort
3.4 - Análise da complexidade do algoritmo quicksort

UNIDADE V - Estruturas de dados dos tipos Árvore Binária e Árvore AVL
6.1 -Árvores, Árvores Binárias e Árvores Binárias de Busca
6.2 -Implementando as Árvores Binária
6.3 - Percorrendo uma Árvore Binária de Busca
6.4 - Percurso em Árvore
6.5 - Inserção
6.6 - Remoção

6.7 - Balanceando uma Árvore
6.7.1 - O Algoritmo DSW
6.7.2 - Árvores AVL

UNIDADE VI - Algoritmos em Grafos
5.1 - Conceitos de grafos.
5.2 - Representação de grafo
5.3 - Algoritmos de busca
5.4 - Algoritmo do caminho mínimo
5.4.1 - Encontrar o melhor caminho do vértice origem ao vértice destino

Procedimentos de Ensino

1. Tipo de aula: teórica e prática

Pode-se trabalhar os 4 tempos da semana reservando 2 tempos para o laboratório e 2 tempos para a sala de aula. A metodologia de ensino deverá ser escolhida levando em consideração a realidade da turma.

2. Aula expositiva dialogada

Nas aulas expositivas dialogadas pode-se tentar reservar 20 ou 30 minutos ao final de cada aula para que os alunos realizem pequenos exercícios e estes sejam corrigidos e discutidos ainda na mesma aula.

Sugestão de como desenvolver:

- Iniciar a aula relembrando quais conceitos foram desenvolvidos nas aulas anteriores;
- Apresentar o que será tratado durante a aula;
- Desenvolver o conteúdo, procurando relacionar com conceitos já apresentados, com exemplos práticos e contextos reais;
- Aplicar exercícios de fixação do aprendizado.

3. Aula prática em laboratório de informática

Não é recomendado levar os alunos para o laboratório para transcrever códigos de livros e material didático, esse tipo de atividade o aluno pode e deve fazer em casa, as aulas de laboratório devem abordar o não-trivial sobre os Algoritmos Avançados que estão sendo vistas, algo que necessite de real intervenção do professor para a plena compreensão pelo aluno. Desafios também podem ser lançados pelo professor à turma de forma a motivá-los.

Sugestão de como desenvolver:

- Utilizar os exercícios apresentados em listas de exercícios como base para desenvolvimento dos programas em laboratório utilizando a linguagem C++.

Recursos

1 - Laboratório de informática;
2 - Equipamento de projeção com computador acoplado;
3 - Software: compilador para a linguagem de programação a ser usada; e
4 - Material para download: listas de exercícios, estudos de caso e resumos.

Procedimentos de Avaliação

O processo de avaliação será composto de três etapas, Avaliação 1 (AV1), Avaliação 2 (AV2) e Avaliação 3 (AV3).

As avaliações poderão ser realizadas através de provas teóricas, provas práticas, e realização de projetos ou outros trabalhos, representando atividades acadêmicas de ensino, de acordo com as especificidades de cada disciplina. A soma de todas as atividades que possam vir a compor o grau final de cada avaliação não poderá ultrapassar o grau máximo de 10, sendo permitido atribuir valor decimal às avaliações. Caso a disciplina, atendendo ao projeto pedagógico de cada curso, além de provas teóricas e/ou práticas contemple outras atividades acadêmicas de ensino, estas não poderão ultrapassar 20% da composição do grau final.

As AV2 e AV3 abrangerão todo o conteúdo da disciplina.

Para aprovação na disciplina o aluno deverá:

1. Atingir resultado igual ou superior a 6,0, calculado a partir da média aritmética entre os graus das avaliações, sendo consideradas apenas as duas maiores notas obtidas dentre as três etapas de avaliação (AV1, AV2 e AV3). A média aritmética obtida será o grau final do aluno na disciplina.
2. Obter grau igual ou superior a 4,0 em, pelo menos, duas das três avaliações.
3. Frequentar, no mínimo, 75% das aulas ministradas.

Bibliografia Básica

CORMEN, Thomas H. **Desmistificando algoritmos**. 1. ed. Rio de Janeiro: Elsevier, 2014.

Bibliografia Complementar

FEOFILOFF, Paulo. **Algoritmos em linguagem C**. 1. ed. Rio de Janeiro: Elsevier, 2009.

KOFFMAN, Elliot B.; WOLFGANG, Paul A. T. **Abstração, Estrutura de Dados e Projeto Usando C++**. Rio de Janeiro: LTC, 2008.

Outras Informações

Algoritmos; CORMEN Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford; Elsevier;