

Aula 01 – Apresentação

Clodoaldo Aparecido de Moraes Lima

26 de fevereiro de 2013

Notas de Aulas baseada na Apostica do Prof. Ivandré e Willian Yukio Honda e notas de aulas da Prof. Graça Nunes (ICMC)

Apresentação

Clodoaldo Aparecido de Moraes Lima

● Formação Acadêmica

- Mestrado Eng. de Automação - UNICAMP
- Doutor em Eng. de Computação - UNICAMP
- Pos-Doutor em Eng. de Computação - UNICAMP

Áreas de Pesquisa

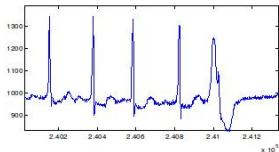
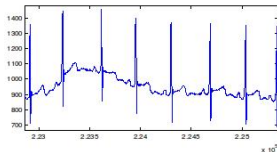
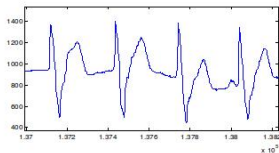
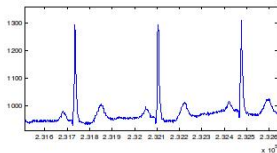
● Áreas de Pesquisa

- Inteligência Computacional
- Aprendizado de Máquina
- Métodos baseados em Kernel
- Sistemas Biométricos
- Análise e Predição de Series Financeira e Biomédicas

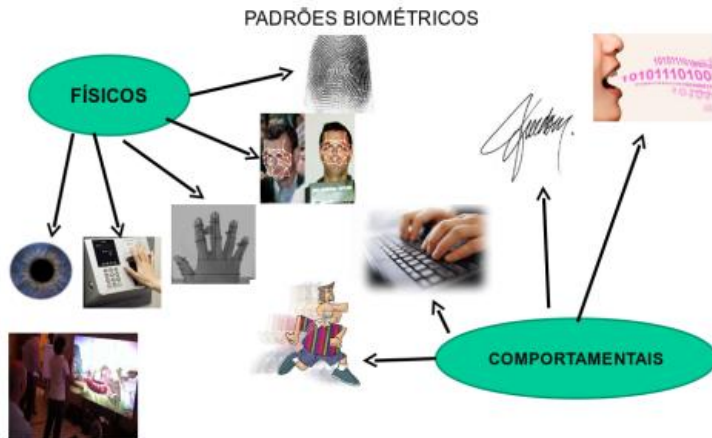
Projetos em Andamento - Mestrado, TCC

- Análise de Sinais Biomédicos - EEG, EMG, ECG
- Sistemas biométricos

Apresentação



Apresentação



Apresentação

Objetivos

- Introduzir conceitos de Estruturas de Dados básicas e seus algoritmos, que são frequentemente usados na construção de programas
- Familiarizar os estudantes com as várias estruturas da informação, buscando habilitá-los a contar com esses recursos no desenvolvimento de outras atividades de ciências de computação.
 - Listas Lineares
 - Listas não lineares
 - Analisar alternativas para sua implementação
 - Construir TAD (Tipo Abstrato de Dados) que possam ser utilizados em outras aplicações

Contéudo Programático

Parte I

- 26/02 – Apresentação da Disciplina
- 01 - Tipos Abstratos de Dados
- 05 e 08/03 - Revisão da Linguagem C
- 12, 15 e 19/3 - Listas Lineares Sequenciais
- 22, 02/4 e 05/4 - Listas Sequenciais - tipos especiais: pilhas e filas
- 09/04, 12/04, 16/04 e 19/04- Listas Lineares Encadeadas Dinâmicas
- 23/04 - Tipos Especiais de Listas
- 26/04 - Revisão
- 30/04 - 1a Prova

Conteúdo Programático

Parte II

- 03/05 - Matrizes Esparsas
- 07/05 - Listas Generalizadas
- 10/05 - Listas Não lineares - Árvores Binárias
- 14/05, 17/5 - Árvore Binária
- 21/05, 24/5 - Árvores de Busca Binária
- 31/05 - Árvore AVL
- 04/06 e 07/06 - Árvore AVL
- 11/06 e 14/06 - Arvore B-tree
- 18/06 - Arvore B-tree
- 21/06 - 2a. Prova
- 25/06 - Prova Substitutiva
- 28/06 - Termina do Semestre

Avaliação

Criterio de Avaliação

- Será realizado duas Provas, peso 4 e 6
- Haverá dois EPs, peso 0.15 e 0.15
- $MediaProvas = (0,4*Prova1+0,6*Prova2)$
- $MediaParcial = 0,15*NotaEP1+0,15*NotaEP2+0,70*MediaProvas$
- Se $MediaProvas \geq 5$, então $NotaFinal = MediaParcial$
- Caso contrario: $NotaFinal = \min(MediaProvas; MediaTemp)$
- Apenas os alunos que faltarem a Prova1 e/ou aa Prova2 terão direito a fazer a prova substitutiva.
- No caso do aluno faltar nas duas provas (Prova1 e Prova2) a nota da prova substitutiva substituirá a nota da Prova2.

Avaliação

Critério e Norma de Recuperação

- Será aprovado todo aluno cuja $\text{NotaFinal} \geq 5$ e presença superior a 70%.
- Os alunos não aprovados pelo critério anterior e que tiveram $\text{NotaFinal} \geq 3$ e apresentarem presença $\geq 70\%$ terão direito a fazer a prova de recuperação.
- Todo aluno que obtiver $\text{NotaRec} \geq 5$ sera aprovado. Neste caso a nota final (NotaFinalRec) sera dada por:
- $\text{NotaFinalRec} = \max(5; (\text{NotaFinal1} + \text{NotaRec})/2)$.
- Se $\text{NotaRec} < 5$, a nota final sera dada por: $\text{NotaFinalRec} = (\text{NotaFinal} + \text{NotaRec})/2$.
- Qualquer tentativa de fraude em provas ou exercicios práticos implicará em zero na disciplina.
- Nas provas, o aluno deverá trazer todo material que precise (caneta e/ou lápis e borracha) e um documento com foto (preferenciamento o cartão USP). As provas tem duração de 90 minutos. Nenhum aluno podera deixar a prova em menos de 15 minutos de seu inicio e nenhum aluno poderá entrar apos a saida de um aluno.

Bibliografia

Basicas

- Apostila: ALGORITMOS E ESTRUTURAS DE DADOS I, Willian Yukio Honda e Ivandre Paraboni
- AHO,A.V.; HOPCROFT,J.E.; ULLMAN,J.D. Data Structure and Algorithms. Readings, Addison Wesley, 1983.
- SCHILDT, HERBERT - C COMPLETO E TOTAL 3a EDICAO, 1997
- HOROWITZ,E.; SAHNI,S. Data Structures in Pascal, Computer Science Press, 1990.
- SZWARCFITER, J.; MARKEZON, L. Estruturas de Dados e seus Algoritmos. LTC Editora, 2a. Ed., 1994.
- WIRTH,N. Algoritmos e Estruturas de Dados, Rio de Janeiro, LTC, 1989.

Bibliografia

Complementar

- GOODRICH, M. T.; TAMASSIA, R., Estruturas de Dados e Algoritmos, Wiley, 2004.
- AHO, A.V.; HOPCROFT, J.E.; ULLMAN, J.D. Data Structure and algorithms. Readings, Addison Wesley, 1982.
- COLLINS, W.J. Data Structures using C And C++, 2nd edition, Prentice-Hall, 1996.
- WEISS, M. A. Data Structures and Algorithm Analysis, The Benjamin/Cummings Pub. Co., 1995.
- WIRTH, N. Algorithms and Data Structures, Englewood Cliffs, Prentice-Hall, 1986.

Compilador

Sugestões de compilador/ambienteCode

- Code::Blocks

- <http://www.codeblocks.org/>

- Dev-C++:

- <http://sourceforge.net/projects/dev-cpp/>

TADs e termos relacionados

Termos relacionados, mas diferentes

- Tipo de dados
- Tipo abstrato de dados - TADs
- Estrutura de dados

TADs e termos relacionados

Tipos de dados

- Em linguagens de programação, o tipo de uma variável define o conjunto de valores que ela pode assumir e como ela pode ser manipulada
 - Por exemplo, uma variável booleana pode ser **true** ou **false**, sendo que operações de AND, OR e NOT podem ser aplicadas sobre ela
- **Novos tipos de dados podem ser definidos em função dos existentes**

Tipos de Dados

Perspectivas

- Computador
 - formas de se interpretar o conteúdo da memória
- Usuário
 - o que pode ser feito em uma linguagem, sem se importar como isso é feito em baixo nível
- Vocês também sentem isso?

Problema

- Como definir um número racional?

Problema

- Como definir um número racional?
- Formas possíveis
 - Um vetor de 2 elementos inteiros, cujo primeiro poderia ser o numerador e o segundo o denominador
 - Um registro de 2 campos inteiros: numerador e denominador
 - Etc.

Variação de Implementação

- Há diferentes implementações possíveis para o mesmo tipo de dado para melhorar
 - Velocidade do código
 - Eficiência em termos de espaço
 - Clareza, etc..
- Todas definem o mesmo domínio e não mudam o significado das operações
 - Para racionais, podemos: criar, somar, multiplicar, verificar se são iguais, imprimir, etc.

Substituição das Implementações

- As mudanças nas implementações têm grande impacto nos programas dos usuários
 - Por exemplo
 - Re-implementação do código
 - Possíveis erros

Exemplo

Programa sobre números racionais

- início
 - declarar i, vetor(10 linhas, 2 colunas) inteiros
 - declarar media real
 - media=0
 - para i=1 até 10 faça
 - ler vetor(i,1) e vetor(i,2)
 - $media = media + \text{vetor}(i,1)/\text{vetor}(i,2)$
 - $media = media/10;$
 - imprimir(media);
 - ∴
- fim

Exemplo

Programa sobre números racionais

- início

- declarar i, vetor(10 linhas, 2 colunas) inteiros
- declarar media real
- media=0
- para i=1 até 10 faça
 - ler vetor(i,1) e vetor(i,2)
 - $media = media + \text{vetor}(i,1)/\text{vetor}(i,2)$
 - $media = media/10;$
- imprimir(media);
- ..

- fim

Qual o impacto de se alterar a forma de representar os números racionais?




Pergunta principal

- Como podemos modificar as implementações dos tipos com o menor impacto possível para os programas que o usam?
- Podemos esconder (encapsular) de quem usa o tipo de dado a forma como foi implementado?

Tipo abstrato de dados

- Tipo de dados **divorciado da implementação**
 - Definido pelo par (v,o)
 - v : valores, dados a serem manipulados
 - o : operações sobre os valores/dados
- Coleção bem definida de dados e um grupo de operadores que podem ser aplicados em tais dados

Tipo abstrato de dados

mun <u>do</u> real	dados de interesse	ESTRUTURA de armazenamento	possíveis OPERAÇÕES
 pessoa	<ul style="list-style-type: none"> a idade da pessoa 	<ul style="list-style-type: none"> tipo inteiro 	<ul style="list-style-type: none"> nasce ($i \leftarrow 0$) aniversário ($i \leftarrow i + 1$)
 cadastro de funcionários	<ul style="list-style-type: none"> o nome, cargo e o salário de cada funcionário 	<ul style="list-style-type: none"> tipo lista ordenada 	<ul style="list-style-type: none"> entra na lista sai da lista altera o cargo altera o salário
 fila de espera	<ul style="list-style-type: none"> nome de cada pessoa e sua posição na fila 	<ul style="list-style-type: none"> tipo fila 	<ul style="list-style-type: none"> sai da fila (o primeiro) entra na fila (no fim)

TAD dicionário inglês-português

- Dados
 - Pares de palavras
- Operações
 - Buscar tradução de uma palavra
 - Inserir novos par de palavras
 - Alteração de informação

Tipo abstrato de dados

- Os dados armazenados podem ser manipulados apenas pelos operadores
 - Ocultamento dos detalhes de representação e implementação, apenas funcionalidade é conhecida
 - Encapsulam dados e comportamento
 - Só se tem acesso às operações de manipulação dos dados, e não aos dados em si

Tipo abstrato de dados e estrutura de dados

- Uma vez que um TAD é definido, escolhe-se a estrutura de dados mais apropriada para representá-lo
 - Exemplos de estruturas de dados

Exemplo

Programa Principal

- programa sobre numeros racionais
- usar TAD de numeros racionais
 - inicio
 - declarar i inteiro
 - declarar r(10) racional
 - para i=1 ate 10 faça
 - ler numeros(r,i)
 - calcular media(r,10)
 - ..
 - fim

TAD de números racional

- Procedimento ler numeros
 - inicio
 - ..
 - fim
- procedimento calcular media
 - inicio
 - ..
 - fim
- ..
-

Exemplo

Programa Principal

- programa sobre numeros racionais
- usar TAD de numeros racionais
 - inicio
 - declarar i inteiro
 - declarar r(10) racional
 - para i=1 ate 10 faça
 - ler numeros(r,i)
 - calcular media(r,10)
 - ..
 - fim

TAD de números racional

- Procedimento ler numeros
 - inicio
 - ..
 - fim
- procedimento calcular media
 - inicio
 - ..
 - fim
- ..
-

Qual o impacto de se alterar a forma de representar os números

Tipo abstrato de dados



Tipo abstrato de dados

Vantagens

- Mais fácil programar
- Não é necessário se preocupar com detalhes de implementação
- Logicamente mais claro
- Mais seguro programar
- Apenas os operadores podem mexer nos dados
- Maior independência, portabilidade e facilidade de manutenção do código
- Maior potencial de reutilização de código
- Abstração
- Conseqüência: custo menor de desenvolvimento

Tipo abstrato de dados

Em termos de implementação, sugerem-se

- Passagem de parametros
 - Um parametro pode especificar um objeto em particular, deixando a operação generica
- Flag para erro, sem emissão de mensagem no codigo principal
 - Independencia do TAD

Tipo abstrato de dados

Em C/C++

- Dados, características, atributos
- Operações, comportamentos, metodos
- Modularidade
- Herança
- Objetos