

## Introdução

SCC502 – Algoritmos e Estruturas de Dados I

## Gerência do curso

- Professores responsáveis
  - Debora Medeiros
    - Sala 3-162
    - [dbr@icmc.usp.br](mailto:dbr@icmc.usp.br)
    - Atendimento: terças, 17h-19h e quartas, 18h-19h
  - Mario Gazziro
    - [mariogazziro@usp.br](mailto:mariogazziro@usp.br)
    - Atendimento: a combinar
- Estagiário PAE
  - Daniel Bonetti
    - Atendimento: a combinar

2

## Avaliação

- 2 provas e 1 SUB (do mal)
  - 26/Set, 28/Nov e 5/Dez
- 2 trabalhos de programação
- Média final
  - $0.7 * \text{média das provas} + 0.3 * \text{média dos projetos}$ , se ambas as médias forem maiores ou iguais a 5; caso contrário, a média final será a menor média dentre elas.
- Frequência mínima requerida: 70% (USP)
- Exercícios realizados durante as aulas podem incrementar a nota do aluno.

3

## Bibliografia básica

- Ziviani, N. (2004). Projeto de Algoritmos com Implementações em Pascal e C. Editora Cengage Learning.
- Mizrahi, V.V. (2008). Treinamento em Linguagem C. Pearson Prentice Hall.
- Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. (2002). Algoritmos: Teoria e Prática. Editora Campus.
- Tenenbaum, A.M.; Langsam, Y.; Augenstein, M.J. (1995). Estruturas de Dados Usando C. Makron Books.

4

## TAD: Tipo Abstrato de Dados (parte 1)

*Baseado no material de Thiago A. S. Pardo*

SCC502 – Algoritmos e Estruturas de Dados I

Debora Medeiros

## TADs e termos relacionados

- Termos relacionados, mas diferentes
  - Tipo de dados
  - Tipo abstrato de dados
  - Estrutura de dados

6

## Tipo 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 elas
- **Novos tipos de dados** podem ser definidos em função dos existentes

7

## Tipo de dados

- Por exemplo, a **declaração de uma variável** específica
  - **Quantidade de bytes** que deve ser reservada a ela
    - Variação entre linguagens
      - Como é em C?
  - Como o dado representado por esses bytes deve ser **interpretado**
    - É inteiro ou real?

8

## Tipo 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
    - Conceito
      - Também sentem isso?

9

## Problema

- Como definir um **número racional**?

10

## Problema

- Como definir um **número racional**?
- Possivelmente como
  - 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.

11

## 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, ver se são iguais, imprimir, etc.

12

## Substituição das implementações

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

13

## 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**?




14

## Tipo abstrato de dados

- Tipo de dados **desvinculado 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**

15

## Tipo abstrato de dados

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

16

## 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

17

## 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?

18

## TAD dicionário inglês-português

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

19

## Exercício

- Fazer um sistema de cadastramento e consulta de pessoas, armazenando o nome e o endereço de cada uma
1. Especificação do TAD
    - Dados/informação
    - Operações
      - Incluir e excluir pessoas do cadastro
      - Dado um nome, achar um endereço
  2. Implementação
    - Representação
    - Código

20

## Operações: exemplo

- Operações de manipulação dos dados
  - está-na-lista?(nome)
  - põe-na-lista(nome,endereço)
  - tira-da-lista(nome)
  - pega-o-endereço(nome)
- Operações de interface (opcional)
  - que-operação-quer-fazer?(operação,nome,end)
  - aí-vai-a-resposta(string-resposta)

21

## Interface: exemplo

```
Repeat forever
que-operação-quer-fazer?( op, nome, endereço )
case op seja...
  "inserir": se está-na-lista?(nome) = verdade
    então ai-vai-a-resposta( 'já está na lista!' )
    senão põe-na-lista( nome, endereço )
    ai-vai-a-resposta( 'ok!' )
  "tirar": se está-na-lista?( nome ) = verdade
    então tira-da-lista( nome )
    ai-vai-a-resposta( 'ok!' )
    senão ai-vai-a-resposta( 'esse cara não está na lista' )
  'acha o endereço': se está-na-lista( nome ) = verdade
    então pega-o-endereço ( nome, endereço )
    ai-vai-a-resposta( 'endereço= ', endereço )
    senão ai-vai-a-resposta( 'não achei o cara!' )
fim do case
fim do repita forever
```

22

## 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

23

## Tipo abstrato de dados

- Em termos de implementação, sugerem-se
  - Passagem de parâmetros
    - Um parâmetro pode especificar um objeto em particular, deixando a operação genérica
  - Flag para erro, sem emissão de mensagem no código principal
    - Independência do TAD

24