

SCC-202 Algoritmos e Estruturas de Dados I

Profa. Graça Nunes
2^o. Semestre 2010

Objetivos

- Introduzir conceitos de Estruturas de Dados básicas e seus algoritmos, que são frequentemente usados na construção de programas
 - Listas Lineares
 - Árvores
- Analisar alternativas para sua implementação
- Construir TAD que possam ser utilizados em outras aplicações

TAD:
Tipo Abstrato de Dados
(parte 1)

SCE-202 – Algoritmos e Estruturas de
Dados I

TADs e termos relacionados

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

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

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?

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?

Problema

- Como definir um número racional?

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.

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.

Substituição das implementações

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




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

mundo real	dados de interesse	ESTRUTURA de armazenamento	possíveis OPERAÇÕES
 <p>pessoa</p>	<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 ($i \leftarrow i + 1$)
 <p>cadastro de funcionários</p>	<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
 <p>fila de espera</p>	<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)

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?

TAD dicionário inglês-português

- Dados

- Pares de palavras

- Operações

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



MUNDO REAL



cadastro de funcionários

operações

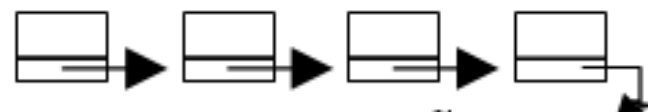
- > inclui alguém no cadastro
- > tira alguém do cadastro
- > altera o cargo
- > altera o salário

FUNCIONALIDADE

```
procedure insere (X : elemento; var L : lista_ordenada);  
  var p, anterior, newnode : lista_ordenada;  
begin  
  p := l; anterior := nil;  
  while (p <> nil) and (p^.info < x) do  
    begin  
      anterior := p;  
      p := p^.next;  
    end;  
  new( newnode );  
  newnode^.info := x;  
  newnode^.next := p;  
  if anterior = nil {x será o primeiro da lista }  
  then L := newnode  
  else anterior^.next := newnode;  
end;
```

IMPLEMENTAÇÃO

Lista →



REPRESENTAÇÃO ABSTRATA

Tipo abstrato de dados

- Em C/C++
 - Dados/características/atributos
 - Operações/comportamentos/métodos
 - Modularidade
 - Herança
 - Objetos

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

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)

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)
- aí-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

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 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