CI1057: Algoritmos e Estruturas de Dados III
Árvores Binárias

Profa. Carmem Hara

Departamento de Informática/UFPR

13 de março de 2024

TAD Árvore Binária - Interface

- `void criaArv( ArvBin* )`:
  cria uma árvore vazia
- `int arvVazia( ArvBin )`:
  retorna 1 se a árvore estiver vazia e zero, caso contrário
- `ArvBin insereArv( Item , ArvBin )`:
  insere um novo item na árvore
- `void escreveArv( ArvBin )`:
  escreve a árvore
- `int alturaArv( ArvBin )`:
  retorna a altura da árvore
- `int contaNoArv( ArvBin )`:
  conta a quantidade de nodos da árvore
- `int arvCompleta( ArvBin )`:
  retorna 1 se a árvore está completa
- `void freeArv( ArvBin )`:
  libera toda a memória alocada para a árvore

Exercício

Utilize as TADs Pilha e ArvBin para percorrer a árvore em pré-ordem.

Entrada: 50 30 70 15 20 80 90 0
Saida: 50 30 15 80 77 90 100
Observe que esta função NÃO pode ser diretamente estendida para precursos em-ordem e pós-ordem. Uma possibilidade para resolver o problem é fazer distinção entre empilhar "árvore" e empilhar "chave".

Exercício

Utilize as TADs Fila e ArvBin para percorrer a árvore por nível.

Entrada: 50 30 70 15 20 80 90 0
Saida: 50 30 80 15 77 90 100

TAD Dicionário

Um **dicionário** ou **tabela de símbolos** é uma estrutura de dados com itens que contém uma *chave* (e opcionalmente um *valor* associado) e que dá suporte a 2 operações básicas:

- *insere*: insere um novo item
- *busca*: retorna o item com uma determinada *chave*

TAD Dicionário - Interface

- `void criaDic( Dic* )`
  cria dicionário vazio
- `int contaItemDic( Dic )`
  conta a quantidade de itens no dicionário
- `int insereDic( Item, *Dic )`
  insere um novo item no dicionário, retorna OK ou ERRO
- `void buscaDic( Chave, Dic, Item* )`
  busca o item com uma determinada chave no dicionario
- `int removeDic( Chave, *Dic )`
  remove o item que contém a chave do dicionário
- `int buscaElementoK( int, Dic )`
  busca o k-ésimo elemento do dicionário
- `void visitaOrdenado( Dic, void (*funcao)(Item) )`
  visita os itens em ordem crescente e aplica a função para cada item visitado

TAD Dicionário - Implementação

- Opção 1: Vetor
- Opção 2: Árvore Binária de Busca

Estrutura da Árvore Binária de Busca

```
1 typedef struct Nodo *ApNodo;
2 typedef struct Nodo {
3   ItemArv item;
4   ApNodo esq, dir;
5   int n;
6 } Nodo;
7
8 typedef ApNodo ArvBusca;
```

Exercício

Implemente as funções para o TAD Árvore de Busca:

- ArvBusca insereArv(ItemArv, ArvBusca)
- ArvBusca insereArvIterativo(ItemArv, ArvBusca)
  Insere um novo Item na árvore como folha - função iterativa
- ArvBusca buscaArv(ItemArv, ArvBusca)
- ArvBusca buscaElementoK(int, ArvBusca)
- void visitaOrdenado(ArvBusca, void (*funcao)(ItemArv))

Verifique o funcionamento das funções utilizando um cliente para o TAD.

Referências

- Secoes 12.1, 12.5 (Sedgewick)
- Capítulo 4 (Sedgewick)