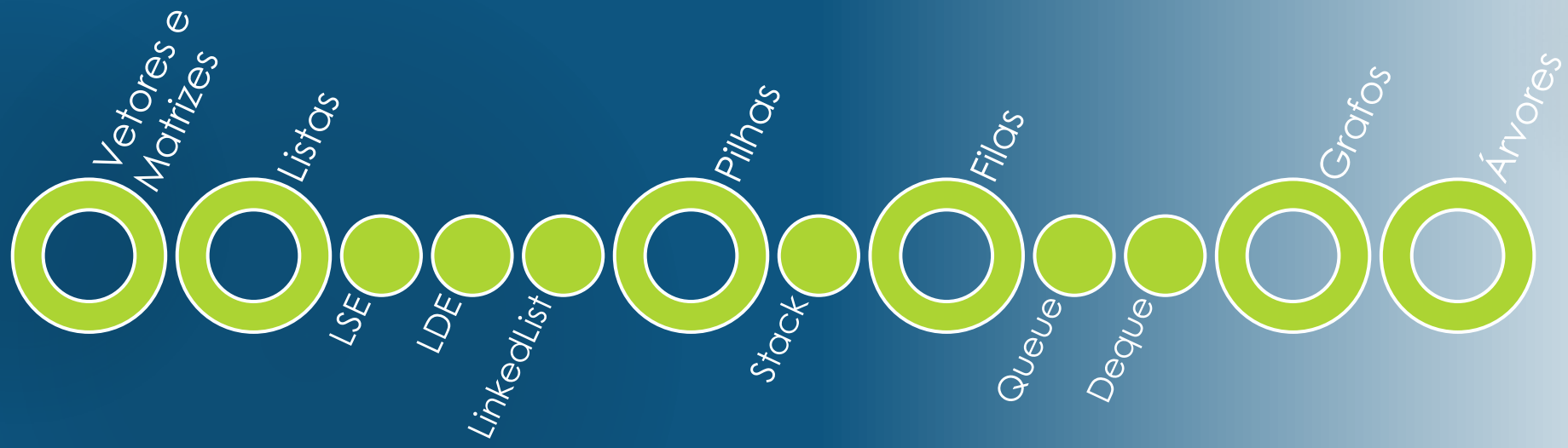
Revisão ED1

PROF. MSC. PAULO CÉSAR MELO

PAULOMELO@INF.UFG.BR

Conteúdo



UFG

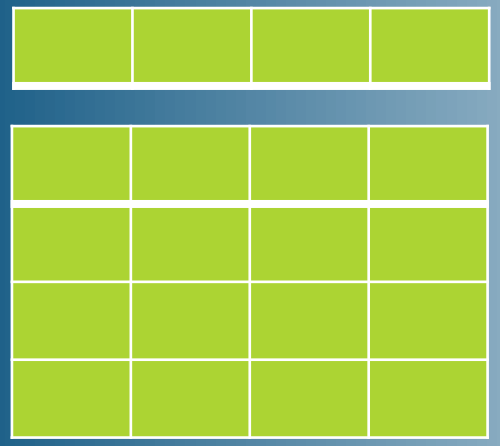
INSTITUTO DE INFORMÁTICA
UFG

Vetores e Matrizes

Vetores e Matrizes

Características

- Alocados em tempo de execução
- Estruturas de alocação estática
- Vetor ➔ unidimensional
- Matriz ➔ bidimensional



Vetores e Matrizes

Representação/API

Vetor

- <tipo> identificador[] = new <tipo>[tamanho];
- String nomes[] = new String[10];

Matriz

- <tipo> identificador[][] = new <tipo> identificador[Tlinha][Tcoluna];
- float notas[][] = new float[10][3].

Vetores e Matrizes

Exercícios

- Faça um programa que contenha métodos que recebam um *array* de inteiros, juntamente com o número de elementos, e calculem:
  - O elemento mais frequente do *array;*
  - A mediana dos elementos no *array* (elemento central), caso tenha tamanho ímpar;
  - A média.

Vetores e Matrizes

Exercícios

- Faça um programa que permita o usuário informar as dimensões para uma matriz nxn (onde n será sempre um número ímpar) e preencha-a da seguinte forma:
  - Diagonal principal: 1
  - Diagonal secundária: 0
  - Elemento central: Y
  - Demais elementos: X

Listas

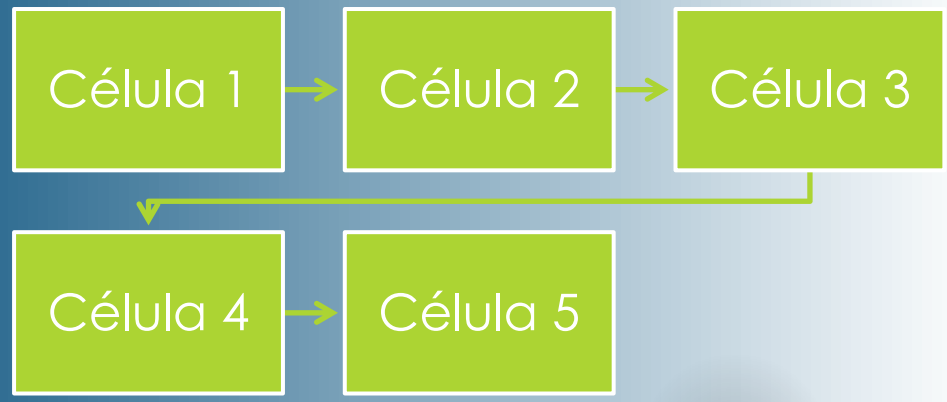
Lista

Características

- Alocação dinâmica
- Manipulação de elementos em qualquer posição da lista

- Lista Simplesmente Encadeada
- Lista Duplamente Encadeada
- Lista Circular

- Célula?



UFG

INSTITUTO DE INFORMÁTICA UFG

Lista

API

- List
- ArrayList
- LinkedList

java.util

Lista

Exercício

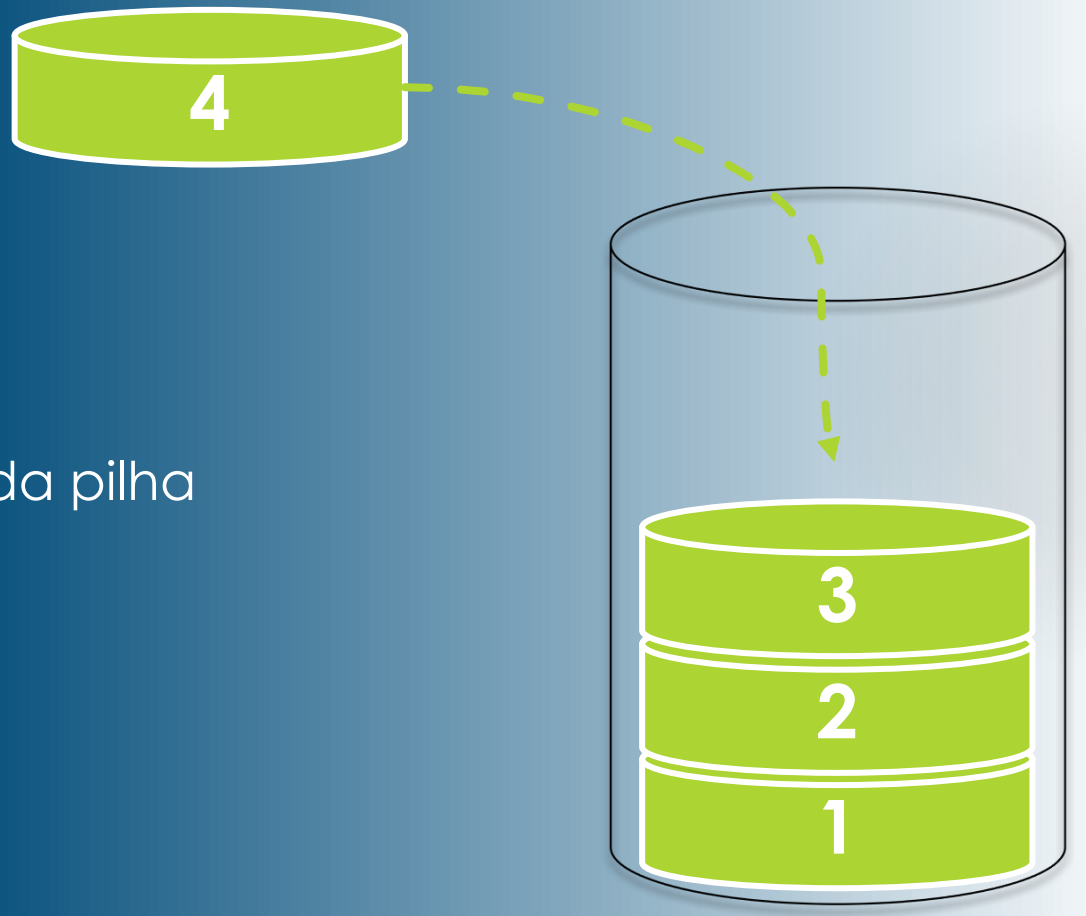
1. Dado uma lista simplesemente ligada com seus elementos dispostos em ordem crescente. Ler um valor X e inseri-lo na lista na posição correta.
   - E não permita que seja inserido elementos repetidos

2. Crie um programa que transfere o maior valor para o início da lista, mantendo os demais na mesma ordem original. Se não for possível fazer a movimentação, imprimir mensagem com o motivo específico, por exemplo: *Lista vazia*, *Lista com um só nó*, *Maior valor já está no início da lista*.

Pilhas e Filas

Pilha

Características



- Estrutura dinâmica
- Acesso somente ao topo da pilha
  - Inserção
  - Remoção
- LIFO
- Uso:
  - Validação de expressão

UFG

INSTITUTO DE INFORMÁTICA
UFG

Pilha

API

- **Stack**

java.util

- Comandos:
  - push
  - pop
  - top (peek)

Pilha

EXERCÍCIO

1. Crie um programa que permita inserir elementos em uma pilha e que o elemento do topo seja sempre a soma dos demais elementos.

2. Dado que duas pilhas de inteiros estão ordenadas crescentemente a partir do topo (topo com o menor valor), crie um programa que transfira os elementos dessas pilhas para uma terceira pilha, inicialmente vazia, de modo que ela fique ordenada decrescentemente com o maior valor no topo.

Fila

Características

- Estrutura dinâmica
- Acesso ao início da fila
- FIFO
- Uso:
  - Fila de impressão

Fila

API

- Queue
  - enqueue/offer
  - dequeue/poll
  - peek
- Deque
  - offer/ offerFirst/ offerLast
  - peek/ peekFirst/ peekLast
  - poll/ poolFirst/ pollLast

Implementa LinkedList

Fila

EXERCÍCIO

1. Crie um programa que gerencie o atendimento de um banco. Este programa deve possuir 3 filas (nesta ordem de prioridade):

   1- Fila para PNE/Idosos/Gestantes;

   2- Fila para correntista;

   3- Fila para não correntista;

Como funcionalidades, este programa deve permitir:

- Cadastrar um usuário (o usuário será um TAD com os atributos, nome, senha, correntista (boolean), atendimento especial (boolean));

- Inserir na fila;

- Realizar atendimento (imprimir a senha e nome do usuário);

- Mostrar lista de atendimentos na ordem em que foram realizados;

Obs.: O atendimento deve ser alternado da seguinte forma:

1º - chamar 1 atendimento especial

2º - chamar 1 correntista

3º - chamar 2 não correntista

UFG

INSTITUTO DE INFORMÁTICA
UFG

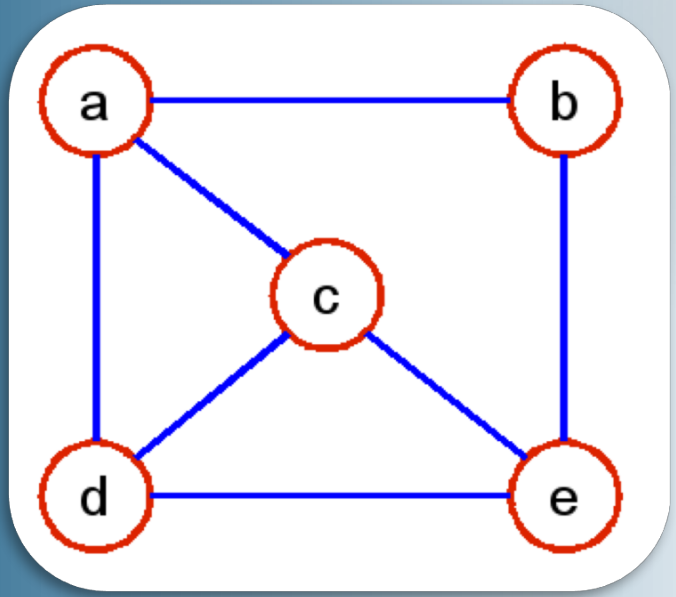
Grafos

Grafo

Características

- Conjunto de vértices e arestas (arcos)

- Estrutura de Dados?
  - Lista de Aresta
  - Lista de Adjacentes
  - Matriz de Adjacentes
  - Matriz de Incidência



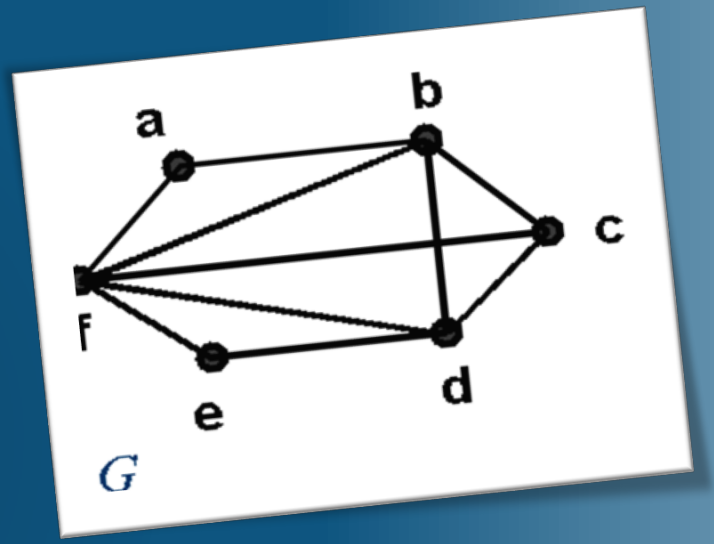
UFG

INSTITUTO DE INFORMÁTICA
UFG

Grafo

EXERCÍCIO

- Represente o grafo abaixo utilizando uma matriz de adjacências e lista de adjacências



- Permita que o usuário digite o grafo (cada nó e seus adjacentes)
- Na saída imprima a matriz de adjacências equivalente e para cada nó, imprima a lista de adjacência.

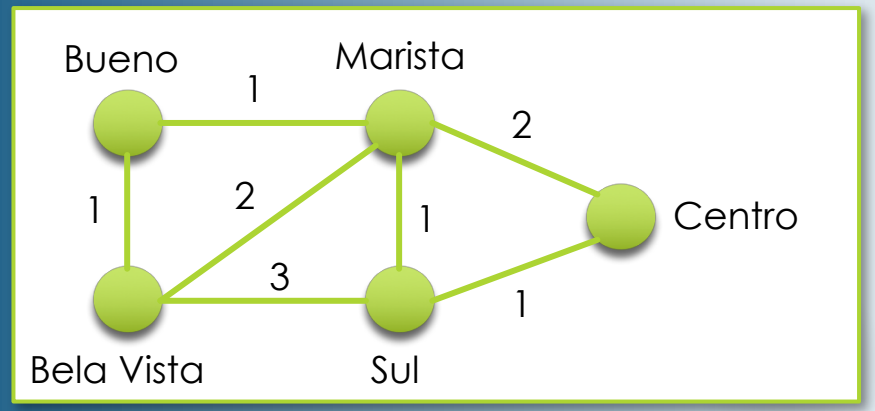
UFG

INSTITUTO DE INFORMÁTICA UFG

Grafo

EXERCÍCIO

- Crie um programa que gere um grafo ponderado, onde o peso de cada aresta represente a distância entre dois setores de Goiânia. O programa deve:
  - Permitir que o usuário insira o gráfico com as distâncias
  - Permitir que o usuário insira o ponto de partida e o destino.
  - Mostrar o todos os possíveis caminhos entre o ponto de partida e o destino escolhido e a distância equivalente.



UFG

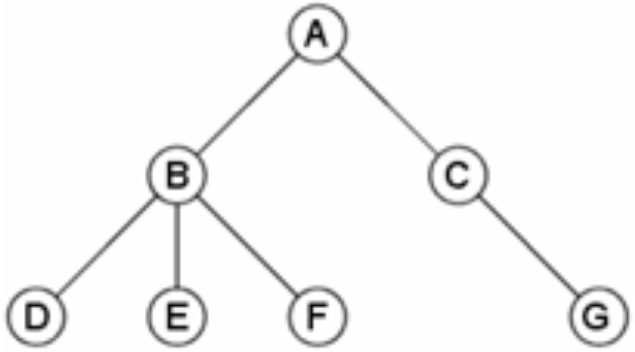
INSTITUTO DE INFORMÁTICA
UFG

Árvores

Árvore

Representação

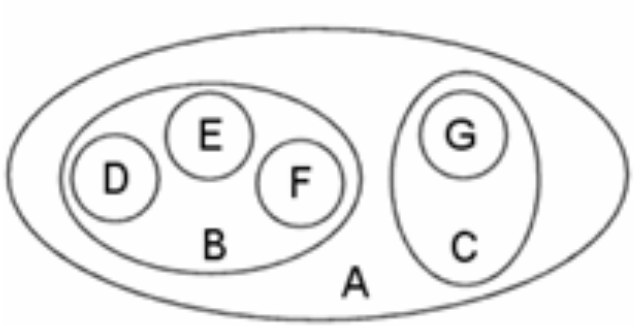
- **Grafo (representação mais utilizada)**



- **Identação**

A
  B
    D
    E
    F
  C
    G

- **Diagrama de Venn** (ou **digrama de inclusão**)



- **Parênteses Aninhados**

(A (B(D, E, F), C(G)))

INSTITUTO DE INFORMÁTICA
UFG

Árvore

Características

- Representar hierarquias
- Raiz
- Filho
- Folha
- Grau
- Nível do nó
- Altura
- Subárvore
- Ancestral e Descendente

Árvore

Caminhamento

- Pré-ordem ➔ um nó é visitado antes de seus descendentes
- Pós-ordem ➔ Um nó é visitado após os seus descendentes
- Em ordem ➔ Percorre a árvore da esquerda para direita

Exemplos

Árvore

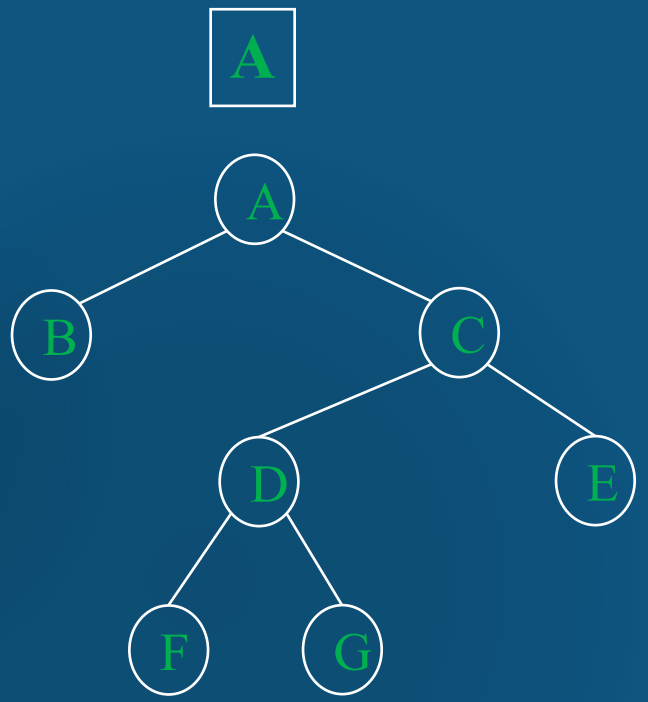
Árvore Binária

- Cada nó, exceto os nós folhas (último nível) podem ter no máximo dois filhos

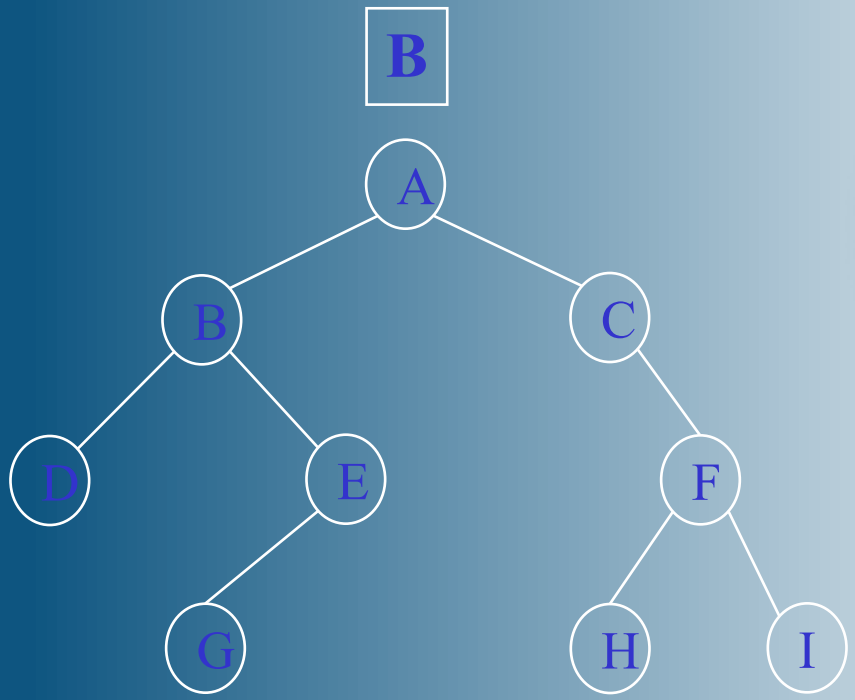
- Completa ➔ Cada nó, exceto os nós folhas (último nível) possui exatamente duas Subárvore

Árvore

Árvore Binária



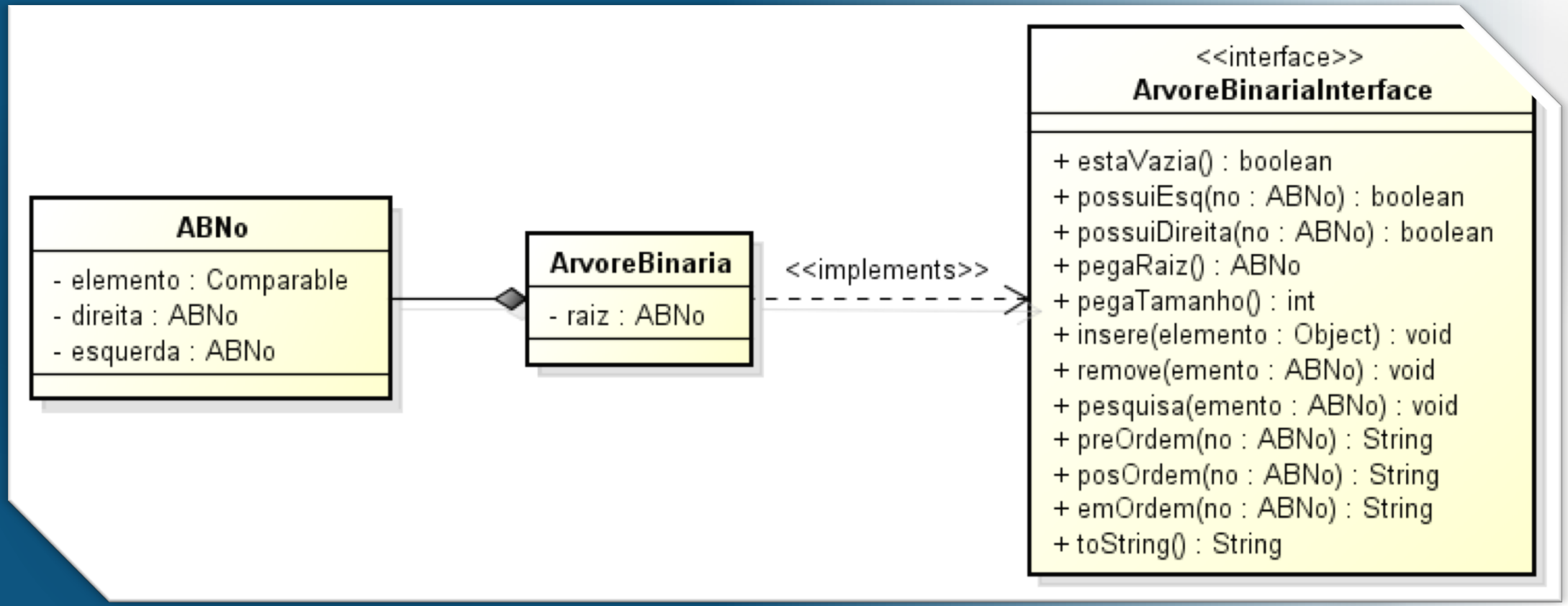
Árvore Binária Completa



Árvore Binária

Árvore

Estrutura de Dados



UFG

INSTITUTO DE INFORMÁTICA
UFG

Árvores - Exercícios

- INSERÇÃO
- PESQUISA
- PRÉ-ORDEM / PÓS-ORDEM / EM ORDEM

Árvore

EXERCÍCIO - Pesquise

- Implemente a pesquisa em uma árvore binária. Utilize o algoritmo abaixo como base.

1. Compare-o com o elemento que está na raiz;
2. Se é menor, vá para a Subárvore esquerda;
3. Se é maior, vá para Subárvore direita;
4. Repita o processo recursivamente, até que a chave procurada seja encontrada ou um nó folha é atingido;
5. Se a pesquisa tiver sucesso então o registro contendo o elemento passada é retornado;

Árvore

EXERCÍCIO - Inserção

- Implemente a inserção de um elemento em uma árvore. Utilize o algoritmo abaixo como base.

1. Verifique se a árvore está vazia;
2. Identificar nó para inserção do elemento (pode usar a pesquisa para auxiliar)

Árvore

EXERCÍCIO – Pré-Ordem

- Implemente o método que percorre a árvore em pré-Ordem e imprima os elementos (nesta ordem). Algoritmo:

1. Visitar a raiz;
2. Caminhar na Subárvore à esquerda, segundo este caminhamento;
3. Caminhar na Subárvore à direita, segundo este caminhamento;

Árvore

EXERCÍCIO – Em Ordem

- Implemente o método que percorre a árvore "em ordem" e imprima os elementos (nesta ordem). Algoritmo:

1. Caminhar na Subárvore à esquerda, segundo este caminhamento;
2. Visitar a raiz;
3. Caminhar na Subárvore à direita, segundo este caminhamento;

Árvore

EXERCÍCIO – Pós-Ordem

- Implemente o método que percorre a árvore em pós-Ordem e imprima os elementos (nesta ordem). Algoritmo:

1. Caminhar na Subárvore à esquerda, segundo este caminhamento;
2. Caminhar na Subárvore à direita, segundo este caminhamento;
3. Visitar a raiz;

#ToDo

Exercícios dos Slides