INSTITUTO FEDERAL
Catarinense
Campus Blumenau

CCA0916 - Estruturas de dados I - BCC

Bacharelado em Ciência da Computação

Prof. Dr. Paulo César Rodacki Gomes

Lista de exercícios - 06 - Árvores binárias

1 Objetivo

O objetivo desta atividade prática em laboratório é implementar árvores binárias conforme visto em sala de aula (os slides estão disponíveis no Goggle Classroom.

2 Implementação orientada a ojetos

Siga as orientações desta seção se você vai implementar sua árvore binária em uma linguagem orientada a objetos. O diagrama de classes da figura 1 mostra a implementação sugerida. Além destas classes, você tambem deve implementar um programa principal para testar e demonstrar sua implementação.

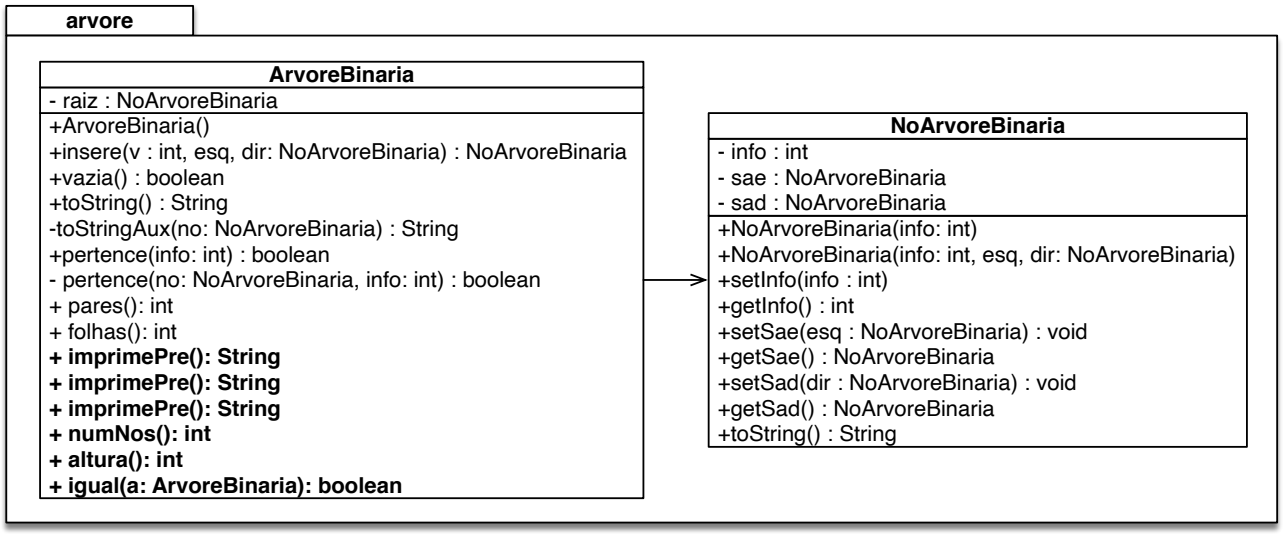

Figura 1: Diagrama de classes do pacote arvore

2.1 Observações importantes

A classe `NoArvoreBinaria` deve ter os seguintes atributos privados:

- `int info`: é o dado armazenado no nó. Para este exercício, vamos considerar que cada nó da árvore armazena um valor do tipo primitivo `int`;
- `NoArvoreBinaria sae` e `sad`: referências para os nós raizes das sub-árvores da esquerda e da direita, respectivamente.

A classe `ArvoreBinaria` deve ter somente o atributo privado `raiz`, que armazena uma referência ao nó raiz principal da árvore.

2.2 Descrição dos métodos a serem implementados

Na sua implementação original de Arvore Binária, acrescente os seguintes métodos (que aparecem em negrito no diagrama de classes da figura 1). Estes métodos necessitam de implementação de método privados auxiliares não especificados na figura 1

1. `insere`: método para inserir um novo valor, juntamente com suas sub-árvores (siga o exemplo em pseudo-código nos slides da aula teórica);
2. `vazia`: este método deve retornar um valor booleano indicando se a árvore é vazia ou não;
3. `toString`: retorna uma string com a representação textual da árvore (pode ser em pré-ordem). No caso de implementação em Python ou C++, pode-se implementar os métodos `__str__` em Python, ou sobrescrever o operador `<<` em C++;
4. `pertence`: verifica se o valor passado como parâmetro está contido na árvore (veja exemplo nos slides);
5. `pares`: retorna a quantidade de nós da árvore que armazenam números pares;
6. `folhas`: retorna a quantidade de nós do tipo folha na árvore;
7. `imprimePre`, `imprimeSim` e `imprimePos`: métodos para imprimir árvores binárias em pré-ordem, ordem simétrica e em pós-ordem, conforme mostrado na aula teórica (hierarquia mostrada por meio do uso de sinais `<` e `>`). O s métodos devem retornar Strings com a representação textual da árvore;
8. `numNos`: retorna a quantidade de nós da árvore;
9. `altura`: retorna a altura da árvore;
10. `igual`: verifica se a árvore atual e a árvore `a` são iguais.

Observação: Após implementar a árvore binária, implemente um programa principal (função ou classe main) para testar e demonstrar o funcionamento da árvore implementada.

3 Implementação em ANSI C

Para implementação em linguagem C, defina uma struct equivalente à classe NoArvoreBinária, juntamente com funções que realizam operações equivalentes àquelas mostradas no diagrama de classes da figura 1
No seu programa principal, a árvore é armazenada como um ponteiro para o nó raiz. Em geral, as funções vão receber esse ponteiro como parâmetro e/ou retornar o endereço do nó, a depender da lógica para implementar a operação correspondente.