PUC Minas

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Curso : Engenharia de *Software*
Disciplina : Algoritmos e Estruturas de Dados II
Professora : Eveline Alonso Veloso

Exercícios sobre Árvores Binárias de Busca

Os exercícios 1 a 8, abaixo, devem ser resolvidos na classe `ABB<E extends Comparable<E>>` implementada durante as aulas teóricas da disciplina.

1) Implemente o método `public void caminhamentoPreOrdem()`, capaz de percorrer a árvore binária de busca em pré-ordem, imprimindo seus elementos. Se a árvore binária de busca estiver vazia, uma exceção deve ser lançada.
   Sugestão: `E` deve sobrescrever a função `public String toString()`, de forma que a classe `ABB<E>` permaneça genérica.

2) Implemente o método `public void caminhamentoPosOrdem()`, capaz de percorrer a árvore binária de busca em pós-ordem, imprimindo seus elementos. Se a árvore binária de busca estiver vazia, uma exceção deve ser lançada.
   Sugestão: `E` deve sobrescrever a função `public String toString()`, de forma que a classe `ABB<E>` permaneça genérica.

3) Implemente o método `public void caminhamentoDecrescente()`, capaz de percorrer a árvore binária de busca, imprimindo seus elementos, em ordem decrescente. A ordem decrescente dos elementos armazenados na árvore binária de busca deve basear-se no(s) critério(s) empregado(s) na implementação do método `public int compareTo(E outroItem)` do item armazenado na ABB.
   Se a árvore binária de busca estiver vazia, uma exceção deve ser lançada.
   Sugestão: `E` deve sobrescrever a função `public String toString()`, de forma que a classe `ABB<E>` permaneça genérica.

4) Implemente o método `public E obterMenor()`, método responsável por recuperar e retornar o menor elemento armazenado na árvore binária de busca. A determinação do menor elemento da árvore binária de busca deve basear-se no(s) critério(s) empregado(s) na implementação do método `public int compareTo(E outroItem)` do item armazenado na ABB.
   Se a árvore binária de busca estiver vazia, uma exceção deve ser lançada.

5) Implemente a função `public ABB<E> clone()`, capaz de criar e retornar uma cópia exata da árvore binária de busca. A árvore binária de busca original e sua cópia, retornada por esse método, podem compartilhar os mesmos elementos. No entanto, elas não devem compartilhar os mesmos nós, ou seja, cada uma das duas árvores binárias de busca deve apresentar seus próprios nós.
   Implemente também, na classe `No<T extends Comparable<T>>` o método `public No<T> clone()`, capaz de criar e retornar uma cópia exata do nó em questão. Utilize o método `clone` do nó para implementar a função `clone` da árvore binária de busca.

6) Implemente a função `public ABB<E> obterSubconjuntoMaiores(E item)`, capaz de criar e retornar um subconjunto da árvore binária de busca formado apenas pelos

elementos da ABB que são maiores ou iguais ao item passado como parâmetro para esse método. A determinação dos elementos da árvore binária de busca que são maiores ou iguais ao item informado como parâmetro para o método deve basear-se no(s) critério(s) empregado(s) na implementação do método `public int compareTo(E outroItem)` do item armazenado na ABB.
Se não for encontrado, na árvore binária de busca, nenhum elemento correspondente ao passado como parâmetro para esse método, uma exceção deve ser lançada.
Sugestão: empregue os métodos `clone`, do nó e da árvore binária de busca, implementados anteriormente.

7) Implemente a função `public boolean ehRaiz(E item)`, capaz verificar e retornar se o item informado como parâmetro para esse método corresponde à raiz da árvore binária de busca. Essa correspondência deve basear-se no(s) critério(s) empregado(s) na implementação do método `equals` do item.
Se a árvore binária de busca estiver vazia, uma exceção deve ser lançada.

8) Implemente o método `public E obterAntecessor(E item)`, capaz de recuperar e retornar o maior elemento armazenado na árvore binária de busca que seja menor do que o item informado como parâmetro para esse método. A determinação dos elementos da árvore binária de busca que são menores ou iguais ao item passado como parâmetro para o método deve basear-se no(s) critério(s) empregado(s) na implementação do método `public int compareTo(E outroItem)` do item armazenado na ABB.
Se não for encontrado, na árvore binária de busca, nenhum elemento correspondente ao passado como parâmetro para esse método, ou o item informado como parâmetro não apresentar antecessor na árvore binária de busca, uma exceção deve ser lançada.