Lista 6 – Árvore Binária de Busca

ESTRUTURA DE DADOS – Pedro Nuno Moura
Monitor: Celio Ferreira Camara Junior

Para as questões abaixo, considere a implementação de Árvore Binária de Busca vista em sala de aula.

ATENÇÃO: Para todas as questões, deve ser informada e explicada a complexidade computacional alcançada.

1) Suponha que uma certa árvore binária de busca possui chaves que são inteiros entre 1 e 10. Deseja-se então buscar pela chave igual a 5. Qual sequência abaixo não pode corresponder à sequência de chaves examinadas na busca pela chave de número 5?

a. 10, 9, 8, 7, 6, 5
b. 4, 10, 8, 7, 5
c. 1, 10, 2, 9, 3, 8, 4, 7, 6, 5
d. 2, 7, 3, 8, 4, 5
e. 1, 2, 10, 4, 8, 5

2) Implemente um método que, dado um `int[] vetor` passado como parâmetro, construa uma árvore binária de busca com os elementos contidos no vetor. O método deve seguir o seguinte protótipo:

```
public void constroiArvore(int[] vetor);
```

3) Duas árvores binárias de busca são similares se: as duas são vazias ou as duas não são vazias, e se suas subárvores da esquerda são similares e se suas subárvores da direita são também similares. Dessa forma, implemente um método de seguinte protótipo para determinar se uma árvore passada como parâmetro é similar à árvore corrente:

```
public boolean eSimilar(ArvBinBusca<T> arvore);
```

4) Implemente um método que, dados parâmetros `Chave chaveMin` e `Chave chaveMax`, remova de uma árvore binária de busca todos os nós cujas chaves não estão contidas no intervalo `[chaveMin, chaveMax]`. O método deve retornar `true` caso tenha sido possível remover ao menos um nó e `false`, caso contrário. Você deve implementar o método com a melhor complexidade possível e deve utilizar o protótipo abaixo:

```
public boolean removeForaIntervalo(Chave chaveMin, Chave chaveMax);
```

5) Implemente um método que, dados dois nós `No primeiroNo` e `No segundoNo` passados como parâmetro, encontre o menor ancestral comum entre esses dois nós da árvore binária de busca. O menor ancestral comum entre esses dois dados nós é aquele de nível mais baixo que tenha ambos `primeiroNo` e `segundoNo` como descendentes. Você deve implementar o método com a melhor complexidade possível e deve adotar o protótipo abaixo:

```
public No obtemAncestralComum(No primeiroNo, No segundoNo);
```

6) Implemente um método que verifique de maneira eficiente se uma árvore binária de busca é balanceada, retornando `true` se a árvore for balanceada e `false`, caso contrário. Uma árvore é balanceada se, e somente se, para todo nó, a diferença absoluta entre as alturas das subárvores esquerda e direita corresponder a 0 ou a 1. Na Figura 1, são exibidas duas árvores binárias, em que uma é balanceada e a outra, não.

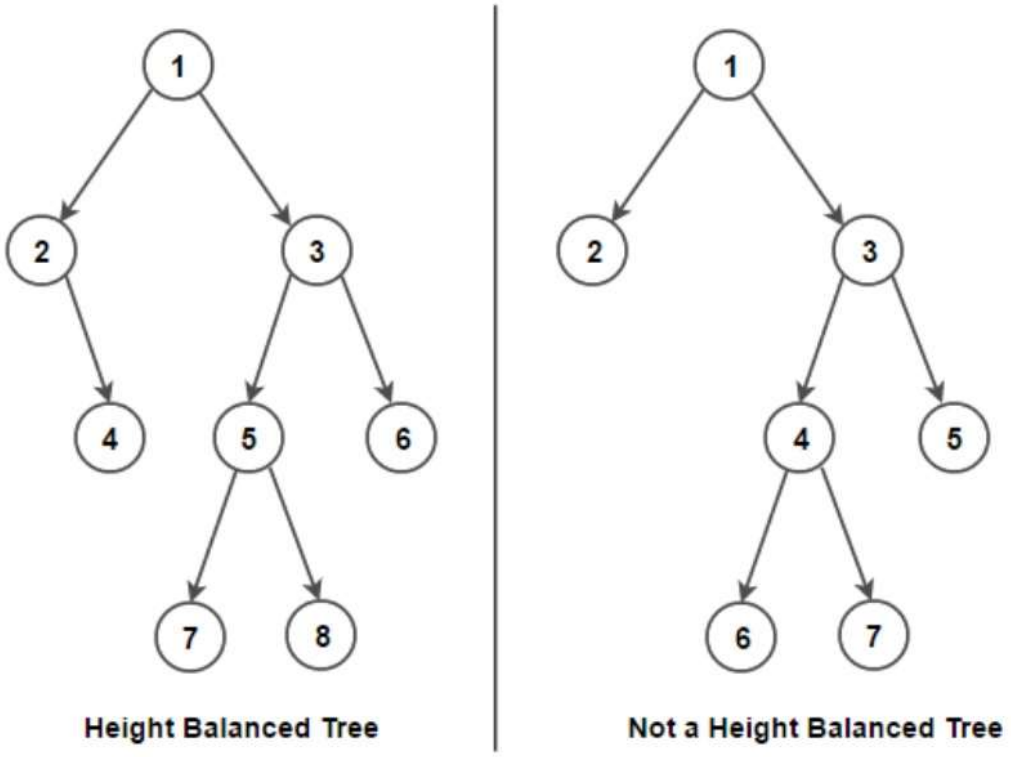

Figura 1: Dois exemplos de árvores binárias em que: a árvore é balanceada (esquerda) e a árvore não é balanceada (direita).

O método a ser implementado deve possuir o protótipo abaixo:

```
public boolean eBalanceada();
```