Árvores (Implentação I)

Antonio Pereira Lima Jr.

Árvore Binária

```
public class BTreeNode {
   // dado do nó
   private int dado;

   // filhos esquerdo e direito
   private BTreeNode esq, dir;

   public BTreeNode() {

   }

   public int getDado() {
      return dado;
    }

   public void setDado(int dado) {
      this.dado = dado;
   }

   public BTreeNode getEsq() {
      return esq;
   }

   public BTreeNode getDir() {
      return dir;
   }

   public void setEsq(BTreeNode nodo) {
      esq = nodo;
   }

   public void setDir(BTreeNode nodo) {
      dir = nodo;
   }
}
```

Árvore Binária

```
public class BSTree {
  private BTreeNode raiz;

  public BSTree() {
     raiz=null;
  }

  public BTreeNode getRaiz() {
     return raiz;
  }
```

Árvore Binária

```
public void inserir(int valor) {
    BTreeNode novo = new BTreeNode(); // cria um novo Nó
    novo.setDado(valor); // atribui o valor recebido ao item de dados do Nó

    if (raiz == null) // árvore vazia, novo será a raiz
    raiz = novo;
    else  {
      BTreeNode atual = raiz;
      BTreeNode anterior;
      while(true) {
        anterior = atual;
        if (valor <= atual.getDado()) { // ir para esquerda
          atual = atual.getEsq();
          if (atual == null) {
            anterior.setEsq(novo);
            return;
          }
        }  // fim da condição ir a esquerda
        else { // ir para direita
           atual = atual.getDir();
           if (atual == null) {
             anterior.setDir(novo);
             return;
           }
        } // fim da condição ir a direita
      } // fim do laço while
    } // fim do else árvore não vazia

  }
```

Árvore Binária

```
  public int altura(BTreeNode atual) {
     if(atual == null)
       return 0;
     else {
   if (altura(atual.getEsq()) > altura(atual.getDir()))
      return ( 1 + altura(atual.getEsq()) );
   else
   return ( 1 + altura(atual.getDir()) );
     }
  }
}
```

Árvore Binária

```
public void inOrder(BTreeNode atual) {
  if (atual != null) {
    inOrder(atual.getEsq());
    System.out.print(atual.getDado() + " ");
    inOrder(atual.getDir());
  }
}

public void preOrder(BTreeNode atual) {
  if (atual != null) {
    System.out.print(atual.getDado() + " ");
    preOrder(atual.getEsq());
    preOrder(atual.getDir());
  }
}

public void posOrder(BTreeNode atual) {
  if (atual != null) {
    posOrder(atual.getEsq());
    posOrder(atual.getDir());
    System.out.print(atual.getDado() + " ");
  }
}
```

Árvore Binária

```
public BTreeNode buscar(int chave) {

  if (raiz == null) return null; // se arvore vazia

  BTreeNode atual = raiz;
  while ((atual.getDado() != chave) & (atual != null)) { // enquanto não encontrou
    if(chave < atual.getDado() )
       atual = atual.getEsq(); // caminha para esquerda
    else
       atual = atual.getDir(); // caminha para direita
  } // fim laço while

  return atual; // se atual é null, chave não encontrada,
                // caso contrário retorna a referência para o nó
}
```

Melhorias?

- Avalie como poderia eliminar as chamadas a métodos getEsq() e getDir() nos métodos da classe BSTree.
  - Dica: modificadores de acesso dos atributos BTreeNode
- Que outras alterações você sugere?
- Pesquise e implemente a remoção de um nó da árvore binária de busca