Classe TreeNode:

package br.unicap.Tree;

class TreeNode<T extends Comparable<T>> {

private T info;

private TreeNode<T> right;

private TreeNode<T> left;

public TreeNode(T info) {

this.info = info;

}

TreeNode<T> insertNode(TreeNode<T> r, T val) {

if (r == null) {

r = new TreeNode(val);

} else if (val.compareTo(r.info) < 0) {

r.setLeft(insertNode(r.getLeft(), val));

} else {

r.setRight(insertNode(r.getRight(), val));

}

return r;

}

TreeNode<T> insertNodeSemRepetidos(TreeNode<T> r, T val) {

if (r == null) {

r = new TreeNode(val);

} else if (val.compareTo(r.info) < 0) {

r.setLeft(insertNodeSemRepetidos(r.getLeft(), val));

} else if (val.compareTo(r.info) > 0){

r.setRight(insertNodeSemRepetidos(r.getRight(), val));

}else{

System.out.println("Repetido!");

}

return r;

}

public T getInfo() {

return info;

}

public void setInfo(T info) {

this.info = info;

}

public TreeNode<T> getRight() {

return right;

}

public void setRight(TreeNode<T> right) {

this.right = right;

}

public TreeNode<T> getLeft() {

return left;

}

public void setLeft(TreeNode<T> left) {

this.left = left;

}

}

Classe Tree:

package br.unicap.Tree;

public class Tree<T extends Comparable<T>> {

private TreeNode<T> root;

public void insert(T val) {

root = this.isEmpty() ? new TreeNode(val) : root.insertNode(root, val);

}

public void insertSemRepetidos(T val){

root = this.isEmpty() ? new TreeNode(val) : root.insertNodeSemRepetidos(root, val);

}

public void insertNaoRecursivoSemRepetidos(T val) {

boolean repetido = false;

TreeNode<T> aux = new TreeNode(val), ant = null, pos;

pos = root;

if (root != null) {

while (pos != null) {

ant = pos;

if (val.compareTo(pos.getInfo()) < 0) {

pos = pos.getLeft();

} else if (val.compareTo(pos.getInfo()) > 0) {

pos = pos.getRight();

} else {

repetido = true;

break;

}

}

if (!repetido) {

if (val.compareTo(ant.getInfo()) < 0) {

ant.setLeft(aux);

} else {

ant.setRight(aux);

}

} else {

System.out.println("Repetido");

}

} else {

root = aux;

}

}

public void exibir() {

System.out.println("PreOrdem");

preordem(this.root);

System.out.println("");

System.out.println("Ordem");

ordem(this.root);

System.out.println("");

System.out.println("PosOrdem");

posordem(this.root);

}

private void preordem(TreeNode r) {

if (r != null) {

System.out.print(r.getInfo() + ", ");

preordem(r.getLeft());

preordem(r.getRight());

}

}

private void posordem(TreeNode r) {

if (r != null) {

posordem(r.getLeft());

posordem(r.getRight());

System.out.print(r.getInfo() + ", ");

}

}

private void ordem(TreeNode r) {

if (r != null) {

ordem(r.getLeft());

System.out.print(r.getInfo() + ", ");

ordem(r.getRight());

}

}

public boolean isEmpty() {

return this.root == null;

}

}