**int** heightDifference(Node node) {

**int** leftTarget = 0;

**int** rightTarget = 0;

**if** (node.left != **null**) {

leftTarget = 1 + node.left.height;

}

**if** (node.right != **null**) {

rightTarget = 1 + node.right.height;

}

**return** leftTarget - rightTarget;

}

**private** Node insert(Node parent, Key key) {

**if** (parent == **null**) **return** **new** Node(key);

**int** cmp = key.compareTo(parent.key);

**if** (cmp <= 0) {

parent.left = insert(parent.left, key);

**if** (heightDifference(parent) == 2) {

**if** (key.compareTo(parent.left.key) <= 0) {

parent = rotateRight(parent);

} **else** {

parent = rotateLeftRight(parent);

}

}

} **else** {

parent.right = insert(parent.right, key);

**if** (heightDifference(parent) == -2) {

**if** (key.compareTo(parent.right.key) > 0) {

parent = rotateLeft(parent);

} **else** {

parent = rotateRightLeft(parent);

}

}

}

computeHeight(parent);

**return** parent;

}

**private** Node rotateRight(Node parent) {

rotations++;

Node newRoot = parent.left;

Node grandson = newRoot.right;

parent.left = grandson;

newRoot.right = parent;

computeHeight(parent);

**return** newRoot;

}