```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

import java.util.*;

class Program {
  // O(n) time | O(n) space - where n is the number of nodes in the Binary Tree
  public static BinaryTree flattenBinaryTree(BinaryTree root) {
    List<BinaryTree> inOrderNodes = getNodesInOrder(root, new ArrayList<BinaryTree>());
    for (int i = 0; i < inOrderNodes.size() - 1; i++) {
      BinaryTree leftNode = inOrderNodes.get(i);
      BinaryTree rightNode = inOrderNodes.get(i + 1);
      leftNode.right = rightNode;
      rightNode.left = leftNode;
    }
    return inOrderNodes.get(0);
  }

  public static List<BinaryTree> getNodesInOrder(BinaryTree tree, List<BinaryTree> array) {
    if (tree != null) {
      getNodesInOrder(tree.left, array);
      array.add(tree);
      getNodesInOrder(tree.right, array);
    }
    return array;
  }

  static class BinaryTree {
    int value;
    BinaryTree left = null;
    BinaryTree right = null;

    public BinaryTree(int value) {
      this.value = value;
    }
  }
}
```