**package bst;**

**class Node {**

**int data;**

**Node left, right;**

**public Node(int item) {**

**data = item;**

**left = right = null;**

**}**

**}**

**public class BST {**

**Node root;**

**// Insertion in BST**

**void insert(int data) {**

**root = insertRec(root, data);**

**}**

**Node insertRec(Node root, int data) {**

**if (root == null) {**

**root = new Node(data);**

**return root;**

**}**

**if (data < root.data) {**

**root.left = insertRec(root.left, data);**

**} else if (data > root.data) {**

**root.right = insertRec(root.right, data);**

**}**

**return root;**

**}**

**// Deletion in BST**

**void delete(int data) {**

**root = deleteRec(root, data);**

**}**

**Node deleteRec(Node root, int data) {**

**if (root == null) {**

**return root;**

**}**

**if (data < root.data) {**

**root.left = deleteRec(root.left, data);**

**} else if (data > root.data) {**

**root.right = deleteRec(root.right, data);**

**} else {**

**if (root.left == null) {**

**return root.right;**

**} else if (root.right == null) {**

**return root.left;**

**}**

**root.data = minValue(root.right);**

**root.right = deleteRec(root.right, root.data);**

**}**

**return root;**

**}**

**int minValue(Node root) {**

**int minValue = root.data;**

**while (root.left != null) {**

**minValue = root.left.data;**

**root = root.left;**

**}**

**return minValue;**

**}**

**// Preorder Traversal**

**void preorderTraversal(Node node) {**

**if (node == null) {**

**return;**

**}**

**System.out.print(node.data + " ");**

**preorderTraversal(node.left);**

**preorderTraversal(node.right);**

**}**

**// Inorder Traversal**

**void inorderTraversal(Node node) {**

**if (node == null) {**

**return;**

**}**

**inorderTraversal(node.left);**

**System.out.print(node.data + " ");**

**inorderTraversal(node.right);**

**}**

**// Postorder Traversal**

**void postorderTraversal(Node node) {**

**if (node == null) {**

**return;**

**}**

**postorderTraversal(node.left);**

**postorderTraversal(node.right);**

**System.out.print(node.data + " ");**

**}**

**public static void main(String[] args) {**

**BST tree = new BST();**

**// Insertion**

**tree.insert(50);**

**tree.insert(30);**

**tree.insert(70);**

**tree.insert(20);**

**tree.insert(40);**

**tree.insert(60);**

**tree.insert(80);**

**System.out.println("Inorder traversal:");**

**tree.inorderTraversal(tree.root);**

**System.out.println("\n\n");**

**// Deletion**

**System.out.println("Delete 30:");**

**tree.delete(30);**

**tree.inorderTraversal(tree.root);**

**System.out.println("\n\n");**

**// Preorder Traversal**

**System.out.println("Preorder traversal:");**

**tree.preorderTraversal(tree.root);**

**System.out.println("\n\n");**

**// Postorder Traversal**

**System.out.println("Postorder traversal:");**

**tree.postorderTraversal(tree.root);**

**}**

**}**