```
package binarytree;

// Java program for different tree traversals

class Node45
{
    int key;
    Node45 left, right;

    public Node45(int item)
    {
        key = item;
        left = right = null;
    }
}

class BinaryTree46
{
    // Root of Binary Tree
    Node45 root;

    BinaryTree46()
    {
        root = null;
    }

    /* Given a binary tree, print its nodes according to the
    "bottom-up" postorder traversal. */
    void printPostorder(Node45 node)
    {
        if (node == null)
            return;

        // first recur on left subtree
        printPostorder(node.left);

        // then recur on right subtree
        printPostorder(node.right);

        // now deal with the node
        System.out.print(node.key + " ");
    }

    /* Given a binary tree, print its nodes in inorder*/
    void printInorder(Node45 node)
    {
        if (node == null)
            return;

        /* first recur on left child */
        printInorder(node.left);

        /* then print the data of node */
        System.out.print(node.key + " ");

        /* now recur on right child */
        printInorder(node.right);
    }

    /* Given a binary tree, print its nodes in preorder*/
    void printPreorder(Node45 node)
    {
        if (node == null)
            return;

        /* first print data of node */
```

```
        System.out.print(node.key + " ");

        /* then recur on left sutree */
        printPreorder(node.left);

        /* now recur on right subtree */
        printPreorder(node.right);
    }

    // Wrappers over above recursive functions
    void printPostorder() {  printPostorder(root); }
    void printInorder() {    printInorder(root); }
    void printPreorder() {   printPreorder(root); }

    // Driver method
    public static void main(String[] args)
    {
        BinaryTree46 tree = new BinaryTree46();
        tree.root = new Node45(1);
        tree.root.left = new Node45(2);
        tree.root.right = new Node45(3);
        tree.root.left.left = new Node45(4);
        tree.root.left.right = new Node45(5);

        System.out.println("Preorder traversal of binary tree is ");
        tree.printPreorder();

        System.out.println("\nInorder traversal of binary tree is ");
        tree.printInorder();

        System.out.println("\nPostorder traversal of binary tree is ");
        tree.printPostorder();
    }
}
```