```java
package binarytree;
/* importing the inbuilt java classes
required for the program */
import java.util.Queue;
import java.util.LinkedList;
// Recursive Java program for level
// order traversal of Binary Tree

/* Class containing left and right child of current
node and key value*/
class Node
{
    int data;
    Node left, right;
    public Node(int item)
    {
        data = item;
        left = right = null;
    }
}

class BinaryTree
{
    // Root of the Binary Tree
    Node root;

    public BinaryTree()
    {
        root = null;
    }

    /* function to print level order traversal of tree*/
    void printLevelOrder()
    {
        int h = height(root);
        int i;
        for (i=1; i<=h; i++)
            printGivenLevel(root, i);
    }

    /* Compute the "height" of a tree -- the number of
    nodes along the longest path from the root node
    down to the farthest leaf node.*/
    int height(Node root)
    {
        if (root == null)
            return 0;
        else
        {
            /* compute height of each subtree */
            int lheight = height(root.left);
            int rheight = height(root.right);

            /* use the larger one */
            if (lheight > rheight)
                return(lheight+1);
            else return(rheight+1);
        }
    }

    /* Print nodes at the given level */
    void printGivenLevel (Node root ,int level)
    {
        if (root == null)
            return;
        if (level == 1)
            System.out.print(root.data + " ");
        else if (level > 1)
        {
            printGivenLevel(root.left, level-1);
            printGivenLevel(root.right, level-1);
```

```
        }
    }

    /* Driver program to test above functions */
    public static void main(String args[])
    {
        BinaryTree tree = new BinaryTree();
        tree.root= new Node(1);
        tree.root.left= new Node(2);
        tree.root.right= new Node(3);
        tree.root.left.left= new Node(4);
        tree.root.left.right= new Node(5);

        System.out.println("Level order traversal of binary tree is ");
                tree.printLevelOrder();
    }
}



// Iterative Queue based Java program
// to do level order traversal
// of Binary Tree




/* Class to print Level Order Traversal */
class BinaryTree99 {

    Node root;

    /* Given a binary tree. Print
    its nodes in level order
    using array for implementing queue */
    void printLevelOrder()
    {
        Queue<Node> queue = new LinkedList<Node>();
        queue.add(root);
        while (!queue.isEmpty())
        {

            /* poll() removes the present head.
            For more information on poll() visit
            http://www.tutorialspoint.com/java/
            util/linkedlist_poll.htm */
            Node tempNode = queue.poll();
            System.out.print(tempNode.data + " ");

            /*Enqueue left child */
            if (tempNode.left != null) {
                queue.add(tempNode.left);
            }

            /*Enqueue right child */
            if (tempNode.right != null) {
                queue.add(tempNode.right);
            }
        }
    }

    public static void main(String args[])
    {
        /* creating a binary tree and entering
        the nodes */
        BinaryTree99 tree_level = new BinaryTree99();
        tree_level.root = new Node(1);
        tree_level.root.left = new Node(2);
        tree_level.root.right = new Node(3);
        tree_level.root.left.left = new Node(4);
```

```
        tree_level.root.left.right = new Node(5);

        System.out.println("Level order traversal of binary tree is - ");
                tree_level.printLevelOrder();
    }
}
```