```java
package binarytree;

// a Binary Tree
// Class to print the Diameter
class BinaryTree35 {
    Node root;

    // Method to calculate the diameter and return it to
    // main
    int diameter(Node root)
    {
        // base case if tree is empty
        if (root == null)
            return 0;

        // get the height of left and right sub-trees
        int lheight = height(root.left);
        int rheight = height(root.right);

        // get the diameter of left and right sub-trees
        int ldiameter = diameter(root.left);
        int rdiameter = diameter(root.right);

        /* Return max of following three
        1) Diameter of left subtree
        2) Diameter of right subtree
        3) Height of left subtree + height of right subtree + 1
        */
        return Math.max(lheight + rheight + 1,
                Math.max(ldiameter, rdiameter));
    }

    // A wrapper over diameter(Node root)
    int diameter() { return diameter(root); }

    // The function Compute the "height" of a tree. Height
    // is the number of nodes along the longest path from the
    // root node down to the farthest leaf node.
    static int height(Node node)
    {
        // base case tree is empty
        if (node == null)
            return 0;

        // If tree is not empty then height = 1 + max of
        // left height and right heights
        return (1
                + Math.max(height(node.left),
                height(node.right)));
    }

    // Driver Code
    public static void main(String args[])
    {
        // creating a binary tree and entering the nodes
        BinaryTree35 tree = new BinaryTree35();
        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);

        // Function Call
        System.out.println(
                "The diameter of given binary tree is : "
                        + tree.diameter());
```

```
    }
}
```