```java
package binarytree;

// Recursive Java program to print lca of two nodes

// A binary tree node


class BinaryTree88
{
    Node root;

    /* Function to find LCA of n1 and n2. The function assumes that both
    n1 and n2 are present in BST */
    Node lca(Node node, int n1, int n2)
    {
        if (node == null)
            return null;

        // If both n1 and n2 are smaller than root, then LCA lies in left
        if (node.data > n1 && node.data > n2)
            return lca(node.left, n1, n2);

        // If both n1 and n2 are greater than root, then LCA lies in right
        if (node.data < n1 && node.data < n2)
            return lca(node.right, n1, n2);

        return node;
    }

    /* Driver program to test lca() */
    public static void main(String args[])
    {
        // Let us construct the BST shown in the above figure
        BinaryTree88 tree = new BinaryTree88();
        tree.root = new Node(20);
        tree.root.left = new Node(8);
        tree.root.right = new Node(22);
        tree.root.left.left = new Node(4);
        tree.root.left.right = new Node(12);
        tree.root.left.right.left = new Node(10);
        tree.root.left.right.right = new Node(14);

        int n1 = 10, n2 = 14;
        Node t = tree.lca(tree.root, n1, n2);
        System.out.println("LCA of " + n1 + " and " + n2 + " is " + t.data);

        n1 = 14;
        n2 = 8;
        t = tree.lca(tree.root, n1, n2);
        System.out.println("LCA of " + n1 + " and " + n2 + " is " + t.data);

        n1 = 10;
        n2 = 22;
        t = tree.lca(tree.root, n1, n2);
        System.out.println("LCA of " + n1 + " and " + n2 + " is " + t.data);

    }
}

// This code has been contributed by Mayank Jaiswal
```