```
import java.util.*;
import java.lang.*;

public class Lca
	{
	public static void main(String arg[])
		{
		int n,i,po,ch,r;
		Scanner in=new Scanner(System.in);

		System.out.println("-----------------------------------------------");
			System.out.println("		LCA OF BINARY TREE");
			System.out.println("-----------------------------------------------");
			System.out.println("Enter the root of the tree");
			r=in.nextInt();
			BT B1=new BT(r);


			System.out.println("-----------------------------------------------");
			System.out.println("		LCA OF BINARY TREE");
			System.out.println("-----------------------------------------------");

			do
			{
			System.out.println("Enter your choice");
			System.out.println("1.		Insert");
			System.out.println("2.		Display Inorder");
			System.out.println("3.		LCA");
			System.out.println("4.		Exit");
			ch =in.nextInt();
			switch(ch)
				{
				case 1:

					System.out.println("Enter position");
					po=in.nextInt();
					B1.root=B1.Insert(B1.root, po);
				break;
				case 2:
					B1.Dis(B1.root);
				break;

				case 3:
				System.out.println("Enter First Node");
					int n1=in.nextInt();
				System.out.println("Enter Second Node");
					int n2=in.nextInt();
					lca(n1,n2,B1.root);
				break;
				}


			}while(ch<4);
		}

	public static void lca(int a,int b,Node p)
		{

		Common_An cl=new Common_An();
		Common_An cr=new Common_An();
```

```
                cl.b=p.l;
                cl.d=a;
                cr.b=p.r;
                cr.d=b;
                cl.start();
                cr.start();
                try
                        {
                        Thread.sleep(1000);
                        }
                catch(InterruptedException e){}

                if((cl.found)&&(cr.found))
                        {
                        System.out.println("Lca is "+ p.data);
                        }

                else if(!(cl.found)&&(cr.found))
                        {
                        lca(a,b,p.r);
                        }

                else if((cl.found)&&!(cr.found))
                        {
                        lca(a,b,p.l);
                        }

                else if(!(cl.found)&&!(cr.found))
                        {
                        lca(b,a,p);
                        }
                }

        }
class Common_An extends Thread
        {
        boolean found=false;
        Node b;
        int d;

        public void run()
                {
                found=Search(b,d);
                System.out.println("Searching "+b.data+" as root");
                }

        public static boolean Search(Node p,int k)
                {
                boolean a=false,b=false;
                if(p!=null)
                        {
                        if(p.data==k)
                                {
                                return true;
                                }
                        a=Search(p.l,k);

                        b=Search(p.r,k);
                        return(a||b);
                        }
```

```
                return(a||b);

                }

        }



class Node
        {
        int data;
        Node l,r,p;

        Node(int n,Node pa)
                {
                p=pa;
                data=n;
                l=null;
                r=null;
                }
        }

class BT
        {
        static Node root;
        BT(int n)
                {
                root=new Node(n,null);
                }



        public static Node Insert(Node p,int po)
                {
                if(p!=null)
                        {
                        Insert( p.l, po);
                        if(p.data==po)
                                {
                                Scanner in=new Scanner(System.in);
                                System.out.println("Enter value");
                                int n= in.nextInt();
                                System.out.println("Enter Left or Right(0/1)");
                                int a= in.nextInt();
                                if(a==0)
                                        {
                                        p.l=new Node(n,p);

                                        }
                                else if(a==1)
                                        {
                                        p.r=new Node(n,p);
                                        }
                                }
                        Insert( p.r, po);
                        return p;
                        }
                        return p;
                }
```

```
public static void Dis(Node p)
        {
        if(p!=null)
                {
                Dis(p.l);
                System.out.println(p.data);
                Dis(p.r);
                }
        }



}
```