```
import java.util.*;

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;
		public static Node currentNode=null,parent=null;

		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 scan=new Scanner(System.in);
						System.out.println("Enter the value:");
						int n=scan.nextInt();
						System.out.println("Enter left or right(0/1):");
						int a=scan.nextInt();
						if(a==0)
						  {
							p.l=new Node(n,p);
						  }
						  else
						  {
							p.r=new Node(n,p);
						  }

					  }
					  Insert(p.r,po);
					  return(p);
				}
				return(p);
		  }

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

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

                        preorder(p.r);
                }
}

public static void postorder(Node p)
{
        if(p!=null)
                {

                        postorder(p.l);

                        postorder(p.r);
                        System.out.println(p.data);
                }
}

public static boolean search(Node p,int n)
        {
                boolean a=false,b=false;

                if(p!=null)
                        {
                                if(p.data==n)
                                  {
                                        parent=p.p;
                                        currentNode=p;
                                        return true;
                                  }
                                a=search(p.l,n);
                                b=search(p.r,n);
                                return(a || b);

                        }
                return(a || b);
        }
public static void delete(Node p)
  {
                 if(p.l==null && p.r==null)
                  {
                          if(parent.l==p)
                                {
                                        parent.l=null;
                                        System.out.println("\nNode "+p.data+" deleted");
                                }
                                else
                                {
                                        parent.r=null;
                                        System.out.println("\nNode "+p.data+" deleted");
                                }
                  }
                  else
                  {
```

```
                                        System.out.println("Not a terminal node");
                                }
                        }
        }

        class bintree
        {
                public static void main(String arg[])
                  {
                        Scanner s=new Scanner(System.in);
                        int ch,no;
                        boolean fl;
                        System.out.println("Enter the root of the tree");
                              int r=s.nextInt();
                              BT B1=new BT(r);
                              do
                              {
                                        System.out.println("\n----------\nBINARY TREE\n(1)INSERT
NODE\n(2)DELETE NODE\n(3)SEARCH NODE");
                                        System.out.println("(4)INORDER TRAVERSAL\n(5)PREORDER
TRAVERSAL\n(6)POSTORDER TRAVERSAL\n(7)EXIT\n");
                                        System.out.println("Enter your choice:");
                                        ch=s.nextInt();
                                        switch(ch)
                                          {
                                                case 1:
                                                        System.out.println("Enter position");
                                                        int po=s.nextInt();
                                                        B1.root=B1.Insert(B1.root, po);
                                                        break;
                                                case 2:  System.out.println("Enter number to delete");
                                                        no=s.nextInt();
                                                        fl=B1.search(B1.root,no);
                                                        if(fl)
                                                         {
                                                                B1.delete(B1.currentNode);

                                                         }
                                                         else
                                                         {
                                                                System.out.println("\n Not Found");
                                                         }
                                                         break;
                                                case 3:  System.out.println("Enter number");
                                                        no=s.nextInt();

                                                        fl=B1.search(B1.root,no);
                                                        if(fl)
                                                         {
                                                                System.out.println("\nFound");
                                                         }
                                                         else
                                                         {
                                                                System.out.println("\n Not Found");
                                                         }
                                                         break;
                                                case 4:         System.out.println("\n Inorder traversal\n");
                                                                B1.inorder(B1.root);
                                                                break;
                                                case 5:         System.out.println("\n Preorder traversal\n");
```

```
                                        B1.preorder(B1.root);
                                        break;
                                case 6:     System.out.println("\n Postorder traversal\n");
                                        B1.postorder(B1.root);
                                        break;

                                case 7: System.exit(0);
                                        break;
                                }
                        }while(ch!=7);
                }
}
```