NAME –KASYAP VARANASI

REGISTRATION NUMBER –20BCE7315

Question- Write a program to implement standard BST operations (insert, delete, search) and also write a method height() to return height of a BST.

Code-

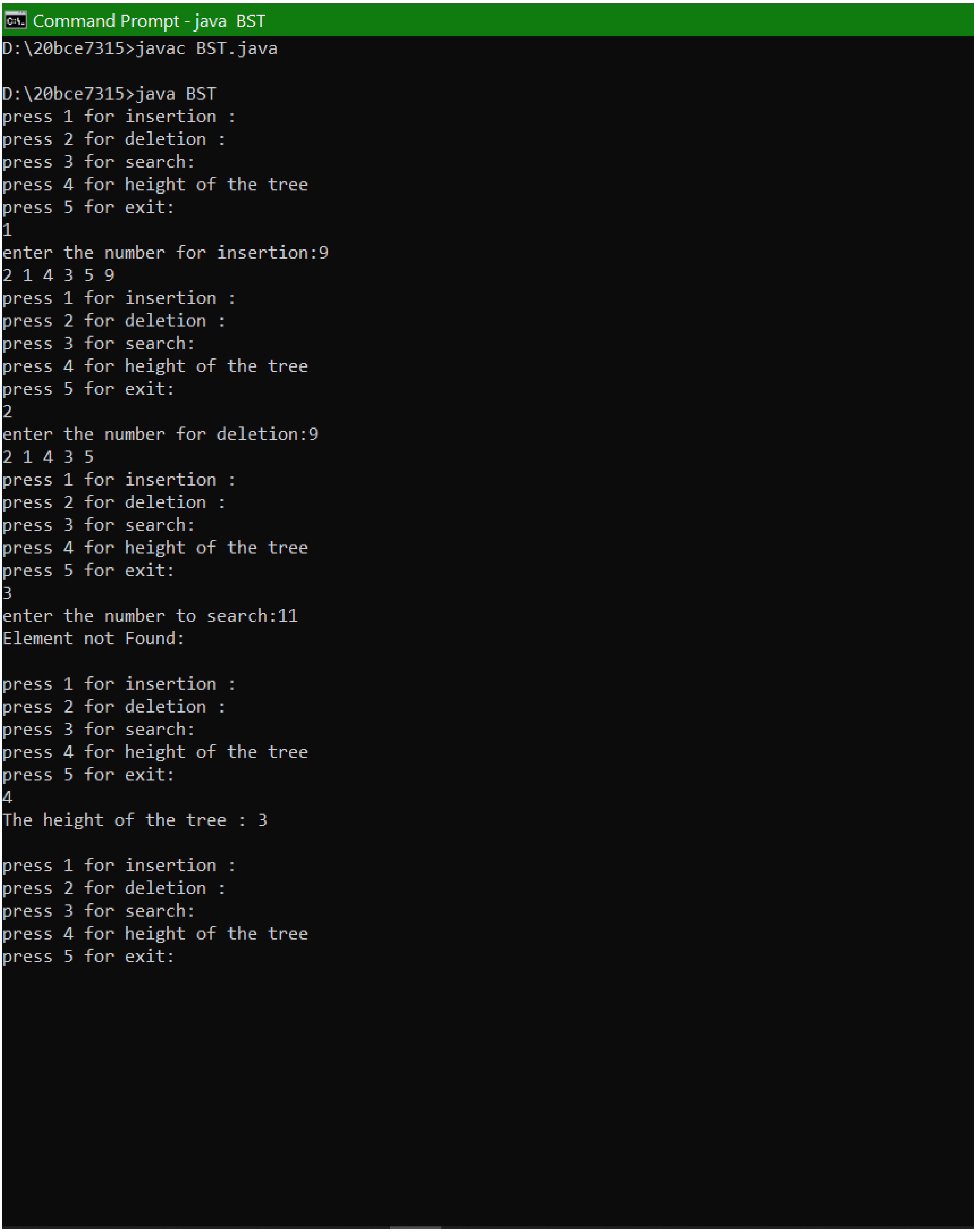
```
import java.util.*;
class Node{
    int key;
    Node left;
    Node right;
    Node (int k){
        key=k;
        left=null;
        right=null;
    }
}
public class BST{
    public static Node insert(Node root, int x) {
        Node temp = new Node(x);
        Node parent = null, curr = root;
        while (curr != null) {
            parent = curr;
            if (curr.key > x)
                curr = curr.left;
            else if (curr.key < x)
                curr = curr.right;
            else
                return root;
        }
        if (parent == null)
            return temp;
        if (parent.key > x)
            parent.left = temp;
        else
            parent.right = temp;
        return root;
    }
    public static Node getSuccessor(Node curr) {
        curr = curr.right;
        while (curr != null && curr.left != null)
            curr = curr.left;
        return curr;
    }

    public static Node delNode(Node root, int x) {
        if (root == null)
            return root;
        if (root.key > x)
            root.left = delNode(root.left, x);
        else if (root.key < x)
            root.right = delNode(root.right, x);
        else {
            if (root.left == null) {
                return root.right;
            } else if (root.right == null) {
                return root.left;
            } else {
                Node succ = getSuccessor(root);
                root.key = succ.key;
                root.right = delNode(root.right, succ.key);
            }
        }
        return root;
    }
    public static boolean search(Node root, int x) {
        while (root != null) {
            if (root.key == x)
                return true;
            else if (root.key < x)
                root = root.right;
            else
                root = root.left;
        }
        return false;
    }
    public static void display(Node root) {
        if (root != null) {
            display(root.left);
            System.out.print(root.key + " ");
            display(root.right);
        }
    }
    public static int maxDepth(Node root)
    {
        if (root == null)
            return 0;
        else
        {
            int lDepth = maxDepth(root.left);
            int rDepth = maxDepth(root.right);
            if (lDepth > rDepth)
                return (lDepth + 1);
             else
                return (rDepth + 1);
        }
    }
    public static void main(String args[]){
```

```
Scanner sc=new Scanner(System.in);
Node root = new Node(1);
root.left = new Node(2);
root.right = new Node(3);
root.right.left = new Node(4);
root.right.right = new Node(5);
int i=0;
int x;
while(i==0)
{
    System.out.println("press 1 for insertion :");
    System.out.println("press 2 for deletion :");
    System.out.println("press 3 for search:");
    System.out.println("press 4 for height of the tree");
    System.out.println("press 5 for exit:");
    int f=sc.nextInt();
    switch(f)
    {
        case 1:System.out.print("enter the number for insertion:");
                x=sc.nextInt();
                root=insert(root,x);
                display(root);
                break;
        case 2: System.out.print("enter the number for deletion:");
                x=sc.nextInt();
                root=delNode(root,x);
                display(root);
                break;
        case 3: System.out.print("enter the number to search:");
                x=sc.nextInt();
                if(search(root,x))
                {
                    System.out.println("Element Found:");
```

```
            }
            else
            {
              System.out.println("Element not Found:");
            }
            break;
        case 4: int a=maxDepth(root);
             System.out.println("The height of the tree : "+a);
             break;
        case 5: i=1;break;
      }
      System.out.println();
    }
  } }
```

Output-



```
D:\20bce7315>javac BST.java

D:\20bce7315>java BST
press 1 for insertion :
press 2 for deletion :
press 3 for search:
press 4 for height of the tree
press 5 for exit:
1
enter the number for insertion:9
2 1 4 3 5 9
press 1 for insertion :
press 2 for deletion :
press 3 for search:
press 4 for height of the tree
press 5 for exit:
2
enter the number for deletion:9
2 1 4 3 5
press 1 for insertion :
press 2 for deletion :
press 3 for search:
press 4 for height of the tree
press 5 for exit:
3
enter the number to search:11
Element not Found:

press 1 for insertion :
press 2 for deletion :
press 3 for search:
press 4 for height of the tree
press 5 for exit:
4
The height of the tree : 3

press 1 for insertion :
press 2 for deletion :
press 3 for search:
press 4 for height of the tree
press 5 for exit:
```