Sukkur IBA University

Data Structure Algorithm

Name: Tariq Mehmood

CMS ID: 023-23-017

Semester: 3rd

Section: E

Submitted by: Sir Riaz Hussain

Lab No: 09

Task 1 and 2

```
class Node{
    int data;
    Node left,right;
    Node(int data){
        this.data=data;
    }
}
public class Task1{
    Node root;
    Node insert(Node root,int data){
        if(root==null){
            root=new Node(data);
            return root;
        }
        if(root.data<data){
            root.right=insert(root.right, data);
        }
        else if(root.data>data){
            root.left=insert(root.left, data);
```

```
        }
        return root;
    }
    boolean search(Node root,int key){
        if(root==null){
            return false;
        }
        if(root.data==key){
            return true;
        }
        if(key<root.data){
            return search(root.left, key);
        }
        else{
            return search(root.right, key);
        }
    }
    Node delete(Node root, int key){
        if(root.data>key){
            root.left=delete(root.left, key);
```

```
}
else if(root.data<key){
    root.right=delete(root.right, key);
}
else{
    //case 1
    if(root.left==null && root.right==null){
        return null;
    }
    //case 2
    if(root.left==null){
        return root.right;
    }
    else if(root.right==null){
        return root.left;
    }
    // Case3
    Node IS=inorderSuccess(root.right);
    root.data=IS.data;
    root.right=delete(root.right, IS.data);
```

```
        }
        return root;
    }
    Node inorderSuccess(Node root){
        while (root.left!=null) {
            root=root.left;
        }
        return root;
    }
    boolean isempty(Node root){
        return root==null;
    }
    void preOrder(Node root){
        if(root==null){
            return;
        }
        System.out.print(root.data+"  ");
        preOrder(root.left);
        preOrder(root.right);
    }
```

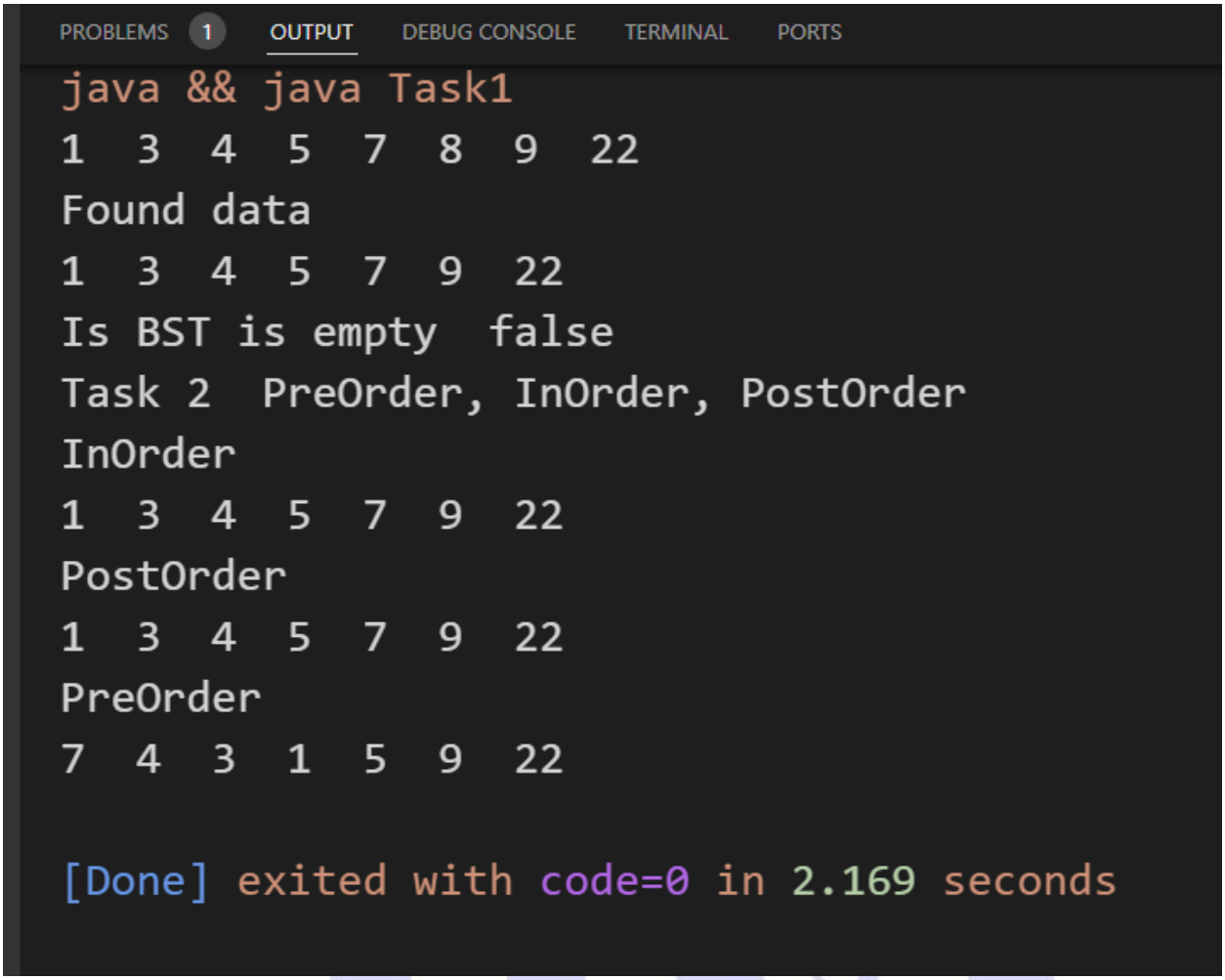
```
void inOrder(Node root){
    if(root==null){
        return;
    }
    inOrder(root.left);
    System.out.print(root.data+"  ");
    inOrder(root.right);
}
void PostOrder(Node root){
    if(root==null){
        return;
    }
    PostOrder(root.left);
    System.out.print(root.data+"  ");
    PostOrder(root.right);
}

public static void main(String[] args) {
    Task1 BST=new Task1();
    Node root=null;
```

```
int values[]={7,4,5,8,9,3,22,1};
for(int i=0; i<values.length; i++){
    root=BST.insert(root, values[i]);
}
// root=BST.insert(root, 05);
// root=BST.insert(root, 06);
// root=BST.insert(root, 03);
BST.inOrder(root);
System.out.println();
boolean check=BST.search(root, 4);
if(check){
    System.out.println("Found data");
}
else{
    System.out.println("Not found ");
}
BST.delete(root, 8);
BST.inOrder(root);
System.out.println();
```

```
        System.out.println("Is BST is empty
"+BST.isempty(root));


        System.out.println("Task 2  PreOrder, InOrder,
PostOrder");
        System.out.println("InOrder ");
        BST.inOrder(root);
        System.out.println();
        System.out.println("PostOrder ");
        BST.PostOrder(root);
        System.out.println();
        System.out.println("PreOrder ");
        BST.preOrder(root);
        System.out.println();
    }
}
```


PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
java && java Task1
1 3 4 5 7 8 9 22
Found data
1 3 4 5 7 9 22
Is BST is empty false
Task 2 PreOrder, InOrder, PostOrder
InOrder
1 3 4 5 7 9 22
PostOrder
1 3 4 5 7 9 22
PreOrder
7 4 3 1 5 9 22
[Done] exited with code=0 in 2.169 seconds

Task 3

```
class Node{
	int data;
	Node left,right;
	Node(int data){
		this.data=data;
	}
}
public class Task2{
	Node root;
	Node insert(Node root,int data){
		if(root==null){
			root=new Node(data);
			return root;
		}
		if(root.data<data){
			root.right=insert(root.right, data);
		}
		else if(root.data>data){
			root.left=insert(root.left, data);
```

```
        }
        return root;
    }
    Node SubtreeNode(Node root, int val){
        if(root==null || root.data==val){
            return root;
        }
        if(val<root.data){
            return SubtreeNode(root.left, val);
        }
        return SubtreeNode(root.right, val);
    }
    void printSubtree(Node root){
        if(root==null){
            return;
        }
        printSubtree(root.left);
        System.out.print(root.data+"  ");
        printSubtree(root.right);
    }
```

```
void inOrder(Node root){
    if(root==null){
        return;
    }
    inOrder(root.left);
    System.out.print(root.data+"  ");
    inOrder(root.right);
}
public static void main(String[] args) {

    Task2 BST=new Task2();
    Node root=null;
    int values[]={7,4,5,8,9,3,22,1};
    /*
            7
        4       9

       3  5        22
      1
```

```
        */
        for(int i=0; i<values.length; i++){
            root=BST.insert(root, values[i]);
        }
        BST.inOrder(root);
        System.out.println();
        Node result=BST.SubtreeNode(root, 4);
        if(result!=null){
            System.out.println("Print  found Node with subtree
"+result.data);
            BST.printSubtree(result);

        }
        else{
            System.out.println("Not found  ");
        }

    }
}
```

```
[Running] cd "d:\BS computer Science\Semester
java && java Task2
1  3  4  5  7  8  9  22
Print  found Node with subtree 4
1  3  4  5
[Done] exited with code=0 in 1.408 seconds
```

Sukkur
IBA