Lab Exam

Name: Onkar Navale

Q1. Perform InOrder tree traversal

Node.java | InOrder.java | InOrderMain.java

```java
package com.tree.oops;

public class Node {

    public int key;
    public Node left;
    public Node right;

    public Node(int key) {
        this.key = key;
        this.left = null;
        this.right = null;
    }

}

```

Node.java | InOrder.java | InOrderMain.java

```java
package com.tree.oops;

public class InOrder {

    public Node root;

    public InOrder() {
        this.root = null;
    }

    public void printInOrder(Node node) {

        if(node == null) {
            return;
        }

        printInOrder(node.left);
        System.out.print(node.key + " ");
        printInOrder(node.right);
    }

    public void printInOrder() {

        printInOrder(root);
    }
}

```

Node.java | InOrder.java | InOrderMain.java

```java
package com.tree.main;

import com.tree.oops.InOrder;

public class InOrderMain {

    public static void main(String[] args) {

        InOrder io = new InOrder();

        io.root = new Node(60);
        io.root.left = new Node(40);
        io.root.right = new Node(80);
        io.root.left.left = new Node(30);
        io.root.left.right = new Node(50);
        io.root.right.left = new Node(70);
        io.root.right.right = new Node(90);

        System.out.println("Inorder traversal: ");
        io.printInOrder();
    }

}
```

Markers | Properties | Servers | Data Source Explorer | Snippets | Terminal | Console

<terminated> InOrderMain [Java Application] E:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win

```
Inorder traversal:
30 40 50 60 70 80 90
```

Q2. Implement stack using array

Stack.java × StackMain.java

```
1  package com.arraystack.oops;
2
3  public class Stack {
4
5      private int[] arr;
6      private int top;
7
8      public Stack(int size) {
9          arr = new int[size];
10         top = -1;
11     }
12
13     public void push(int element) {
14
15         if(top == arr.length-1) {
16             System.out.println("Stack is full, wont be able to push element");
17             return;
18         }
19         top = top + 1;
20         arr[top] = element;
21         System.out.println("element pushed into stack");
22     }
23
24     public Integer pop() {
25
26         if( top == -1) {
27             return null;
28         }
29         int element = arr[top];
30         top--;
31         return element;
32     }
```

```
31         return element;
32     }
33
34     public Integer peek() {
35         if (top == -1) {
36             return null;
37         }
38         return arr[top];
39     }
40
41     public Integer count() {
42
43         return top+1;
44     }
45
46     public void displayStack() {
47         for(int i = top; i >= 0; i--) {
48             System.out.print(arr[i] + " ");
49         }
50     }
51 }
52
```

Stack.java | StackMain.java

```java
package com.arraystack.main;

import com.arraystack.oops.Stack;

public class StackMain {

    public static void main(String[] args) {

        Stack s = new Stack(4);

        s.push(12);
        s.push(23);
        s.push(45);
        s.push(10);

        System.out.println();
        System.out.println("All elements of stack are: ");
        s.displayStack();

        System.out.println();
        System.out.println();
        System.out.println("Popped element is: " + s.pop());

        System.out.println();
        System.out.println("New elements of stack are: ");
        s.displayStack();

        System.out.println();
        System.out.println();
        System.out.println("Element using peek method: " + s.peek());

        System.out.println();
        System.out.println("Number of elements are : " + s.count());
    }

}
```

Markers | Properties | Servers | Data Source Explorer | Snippets | Terminal | Console

<terminated> StackMain (1) [Java Application] E:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot

```
element pushed into stack
element pushed into stack
element pushed into stack
element pushed into stack

All elements of stack are:
10 45 23 12

Popped element is: 10

New elements of stack are:
45 23 12

Element using peek method: 45

Number of elements are : 3
```