Exp No 2

Program:

```
//AddInterface.java package com.saif.exp2;

import java.rmi.*;
public interface AddInterface extends Remote {
public int sum(int nl, int n2) throws RemoteException;

}

//Add.java
package com.saif.exp2;

import java.rmi.*; import java.rmi.server.*;
public class Add extends UnicastRemoteObject implements AddInterface { int num1, num2;
public Add() throws RemoteException {
}
public int sum(int n1, int n2) throws RemoteException { num1 = n1;
num2 = n2;
return num1 + num2;
}
}

//AddServer.java package com.saif.exp2;

import java.rmi.Naming; public class AddServer {

public static void main(String[] args) { try {
Naming.rebind("Add", new Add());
System.out.println("Server is connected and waiting for the client");
} catch (Exception e) {
```

System.out.println("Server could not connect: " + e);

}

}

}

//AddClient.java package com.saif.exp2;

import java.rmi.Naming; public class AddClient {

public static void main(String[] args) { try

AddInterface ai = (AddInterface) Naming.*lookup*("//localhost/Add"); System.*out*.println("The sum of 2 numbers is: " + ai.sum(10, 2));

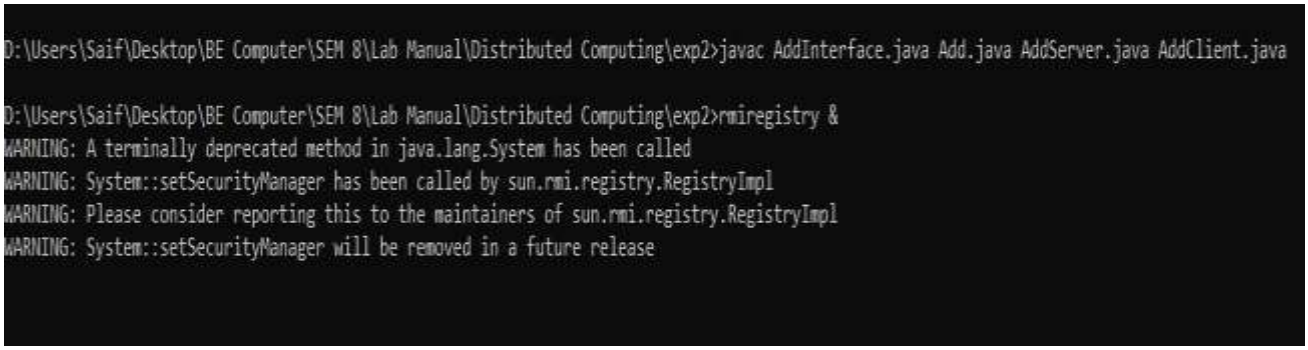
} catch (Exception e) { System.*out*.println("Client Exception: " + e);

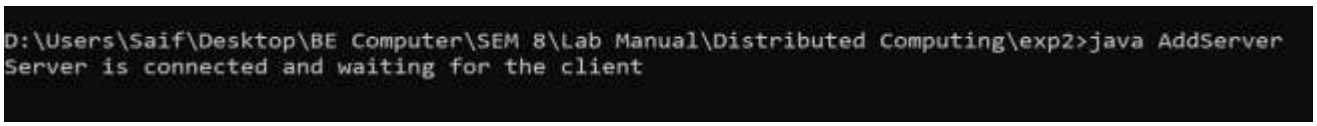
}

}

}

Output:





Exp 3

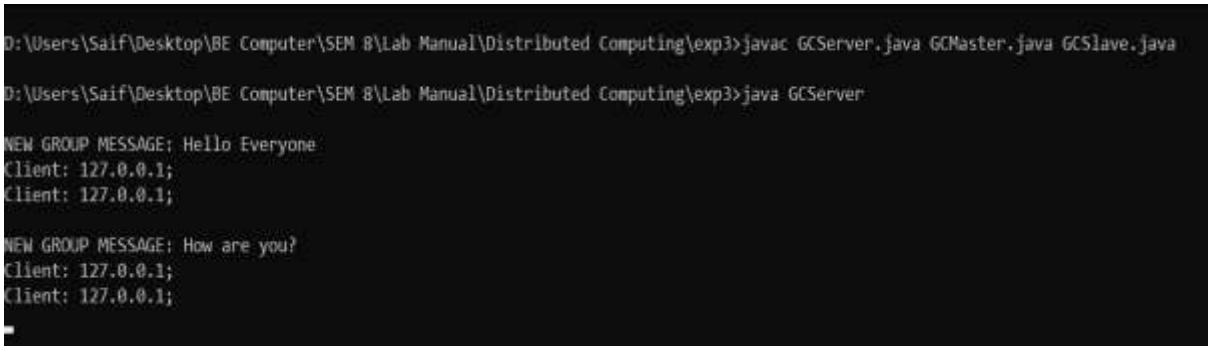
Program:

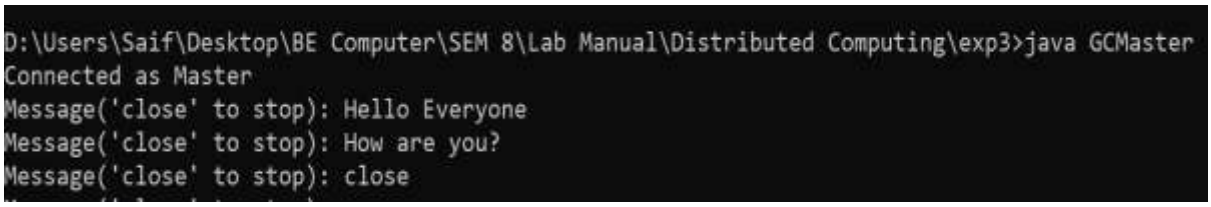
```
//GCServer.java package com.saif.exp3;

import java.io.*; import java.util.*; import java.io.*; import java.net.*;

public class GCServer {

static ArrayList<ClientHandler> clients = new ArrayList<ClientHandler>();

public static void main(String[] args) throws Exception { ServerSocket server = new ServerSocket(25);
Message msg = new Message();

int count = 0; while (true) {

Socket ss = server.accept();

DataInputStream din = new DataInputStream(ss.getInputStream()); DataOutputStream dout = new
DataOutputStream(ss.getOutputStream()); ClientHandler chlr = new ClientHandler(ss, din, dout,
msg);

                Thread t = chlr; clients.add(chlr); count++; t.start();}}}

class Message { String msg;

public void setMsg(String msg) { this.msg = msg;

}

public void getMsg() {

System.out.println("\nNEW GROUP MESSAGE: " + this.msg); for (int i = 0; i < GCServer.clients.size();
i++) {

try {

System.out.println("Client: " + GCServer.clients.get(i).ip + "; ");
GCServer.clients.get(i).out.writeUTF(this.msg); GCServer.clients.get(i).out.flush();

} catch (Exception e) { System.out.print(e);}}}}

class ClientHandler extends Thread { DataInputStream in; DataOutputStream out;

Socket socket; int sum;

float res; boolean conn; Message msg; String ip;


public ClientHandler(Socket s, DataInputStream din, DataOutputStream dout, Message msg) {
this.socket = s;

this.in = din; this.out = dout; this.conn = true; this.msg = msg;
```

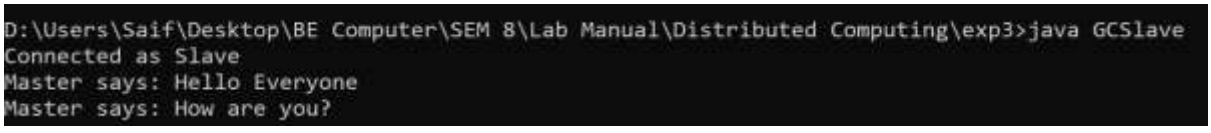
```
this.ip = (((InetSocketAddress)
this.socket.getRemoteSocketAddress()).getAddress()).toString().replace("/", "");
}
public void run() {
while (conn == true) { try {
String input = this.in.readUTF(); this.msg.setMsg(input); this.msg.getMsg();
} catch (Exception e) {
conn = false; System.out.println(e);}}
closeConn();
}
public void closeConn() { try {
this.out.close(); this.in.close(); this.socket.close();
} catch (Exception e) { System.out.println(e);
}
}
}
//GCMaster.java package com.saif.exp3;
import java.util.*; import java.io.*; import java.net.*;
public class GCMaster {
public static void main(String[] args) throws Exception { Socket client = new Socket("127.0.0.1", 25);
DataInputStream din = new DataInputStream(client.getInputStream()); DataOutputStream dout =
new DataOutputStream(client.getOutputStream()); System.out.println("Connected as Master");
Scanner sc = new Scanner(System.in); String send = "";
do {
System.out.print("Message('close' to stop): "); send = sc.nextLine();
dout.writeUTF(send); dout.flush();
} while (!send.equals("stop")); dout.close();
din.close(); client.close();
}
}

//GCSlave.java package com.saif.exp3;
```

```
import java.io.DataInputStream; import java.net.Socket;


public class GCSlave {

public static void main(String[] args) throws Exception{ Socket client = new Socket("127.0.0.1",25);

DataInputStream din = new DataInputStream(client.getInputStream());
System.out.println("Connected as Slave");

String recv = ""; do{

recv = din.readUTF(); System.out.println("Master says: " + recv);

}while(!recv.equals("stop")); din.close();

client.close();

}

}
```

Output:







Exp 4

Program:

//Lamport.java

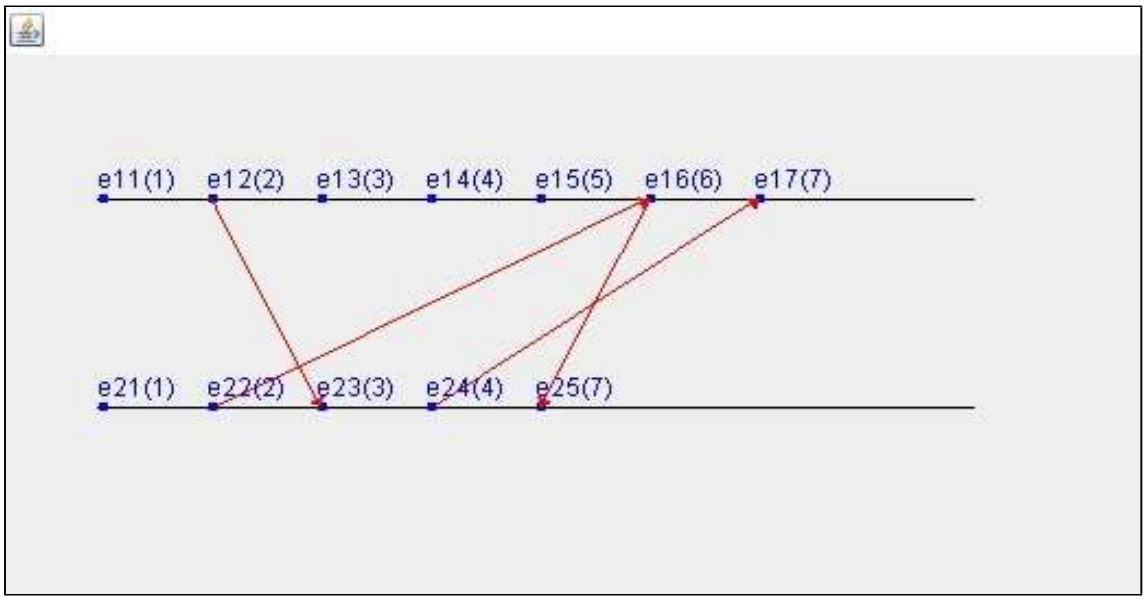
```
import java.util.*;
import java.util.HashMap; import java.util.Scanner; import javax.swing.*;
import java.awt.*;
import java.awt.geom.*; public class Lamport {
int e[][] = new int[10][10];
int en[][] = new int[10][10]; int ev[] = new int[10];
int i, p, j, k;

HashMap<Integer, Integer> hm = new HashMap<Integer, Integer>(); int
xpoints[] = new int[5];
int ypoints[] = new int[5];

class draw extends JFrame { private final int ARR_SIZE = 4;
void drawArrow(Graphics g1, int x1, int y1, int x2, int y2) { Graphics2D g =
(Graphics2D) g1.create();
double dx = x2 - x1, dy = y2 - y1; double angle = Math.atan2(dy, dx);
int len = (int) Math.sqrt(dx * dx + dy * dy);
AffineTransform at = AffineTransform.getTranslateInstance(x1, y1);
at.concatenate(AffineTransform.getRotateInstance(angle)); g.transform(at);
// Draw horizontal arrow starting in (0,0)

g.drawLine(0, 0, len, 0);
```

```
g.fillPolygon(new int[]{len, len - ARR_SIZE, len - ARR_SIZE, len}, new int[]{0, -
ARR_SIZE, ARR_SIZE, 0}, 4);
}

public void paintComponent(Graphics g) { for (int x = 15; x < 200; x += 16) {
drawArrow(g, x, x, x, 150); drawArrow(g, 30, 300, 300, 190);
}
23

}
public void paint(Graphics g) { int h1, h11, h12;
Graphics2D go = (Graphics2D) g; go.setPaint(Color.black);
for (i = 1; i <= p; i++) {
go.drawLine(50, 100 * i, 450, 100 * i);
}

for (i = 1; i <= p; i++) {

for (j = 1; j <= ev[i]; j++) { k = i * 10 + j; go.setPaint(Color.blue);
go.fillOval(50 * j, 100 * i - 3, 5, 5);
go.drawString("e" + i + j + "(" + en[i][j] + ")", 50 * j, 100 * i - 5); h1 = hm.get(k);
if (h1 != 0) { h11 = h1 / 10;
h12 = h1 % 10;
go.setPaint(Color.red);
drawArrow(go, 50 * h12 + 2, 100 * h11, 50 * j + 2, 100 * i);
}}}}}
public void calc() {
```

```
Scanner sc = new Scanner(System.in);
System.out.println("Enter the number of process:"); p = sc.nextInt();
System.out.println("Enter the no of events per process:"); for (i = 1; i <= p; i++) {
ev[i] = sc.nextInt();}
System.out.println("Enter the relationship:");
for (i = 1; i <= p; i++) { System.out.println("For process:" + i); for (j = 1; j <= ev[i]; j++) {
System.out.println("For event:" + (j)); int input = sc.nextInt();
k = i * 10 + j; hm.put(k, input); if (j == 1) {
en[i][j] = 1;}}}
for (i = 1; i <= p; i++) {
for (j = 2; j <= ev[i]; j++) { k = i * 10 + j;
if (hm.get(k) == 0) { en[i][j] = en[i][j - 1] + 1;
} else {
int a = hm.get(k); int p1 = a / 10; int e1 = a % 10;
if (en[p1][e1] > en[i][j - 1]) {
en[i][j] = en[p1][e1] + 1;
} else {
en[i][j] = en[i][j - 1] + 1;}}}}
for (i = 1; i <= p; i++) {
for (j = 1; j <= ev[i]; j++) {
System.out.println(en[i][j]);}}
JFrame jf = new draw(); jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); jf.setSize(500,500);
jf.setVisible(true);}
public static void main(String[] args) {
Lamport lam = new Lamport(); lam.calc();}}
```

Output:

```
C:\Users\Saif Bodu\IdeaProjects\DC_Experiments\src\com\saif\exp4>javac Lamport.java

C:\Users\Saif Bodu\IdeaProjects\DC_Experiments\src\com\saif\exp4>java Lamport
Enter the number of process:
2
Enter the no of events per process:
7
5
Enter the relationship:
For process:1
For event:1
0
For event:2
0
For event:3
0
For event:4
0
For event:5
0
For event:6
22
For event:7
24
For process:2
For event:1
0
For event:2
0
For event:3
12
For event:4
0
For event:5
16
1
2
3
4
5
6
7
1
2
3
4
7
```



Program:

```
//Bully.java

package com.saif.exp5;

import java.io.*; import java.util.*; public class Bully {

static int n;

static int pro[] = new int[100]; static int sta[] = new int[100]; static int co;

public static void main(String[] args) { System.out.print("Enter the number of process: ");
Scanner sc = new Scanner(System.in);

n = sc.nextInt(); int i, j, c, cl = 1;

for (i = 0; i < n; i++) {

sta[i] = 1;

pro[i] = i;}

boolean choice = true; int ch;

do {

System.out.println("Enter Your Choice"); System.out.println("1. Crash Process");
System.out.println("2. Recover Process"); System.out.println("3. Exit"); System.out.print("> ");

ch = sc.nextInt(); switch (ch) {

case 1:

System.out.print("Enter the process number: "); c = sc.nextInt();

sta[c - 1] = 0;

cl = 1; break;

case 2:

System.out.print("Enter the process number: "); c = sc.nextInt();

sta[c - 1] = 1;

cl = 1; break;

case 3:

choice = false; cl = 0;


break;}
```

```
if (cl == 1) {

System.out.print("Which process will initiate election? = "); int ele =
sc.nextInt();

elect(ele);}

System.out.println("Final coordinator is " + co);} while (choice);}

static void elect(int ele) { ele = ele - 1;co = ele + 1;

for (int i = 0; i < n; i++) { if (pro[ele] < pro[i]) {

System.out.println("Election message is sent from " + (ele + 1) + " to " + (i + 1));
if (sta[i] == 1) {

System.out.println("Ok message is sent from " + (i + 1) + " to " + (ele + 1));}

if (sta[i] == 1) {elect(i + 1);

}}}}}
```

Output:

```
D:\Users\Saif\Desktop\BE Computer\SEM 8\Lab Manual\Distributed Computing\exp5>javac Bully.java && java Bully
Enter the number of process: 5
Enter Your Choice
1. Crash Process
2. Recover Process
3. Exit
> 1
Enter the process number: 1
Which process will initiate election? = 2
Election message is sent from 2 to 3
Ok message is sent from 3 to 2
Election message is sent from 3 to 4
Ok message is sent from 4 to 3
Election message is sent from 4 to 5
Ok message is sent from 5 to 4
Election message is sent from 3 to 5
Ok message is sent from 5 to 3
Election message is sent from 2 to 4
Ok message is sent from 4 to 2
Election message is sent from 4 to 5
Ok message is sent from 5 to 4
Election message is sent from 2 to 5
Ok message is sent from 5 to 2
Final coordinator is 5
Enter Your Choice
1. Crash Process
2. Recover Process
3. Exit
> 1
Enter the process number: 5
Which process will initiate election? = 3
Election message is sent from 3 to 4
Ok message is sent from 4 to 3
Election message is sent from 4 to 5
Election message is sent from 3 to 5
Final coordinator is 4
Enter Your Choice
1. Crash Process
2. Recover Process
3. Exit
> 2
Enter the process number: 1
Which process will initiate election? = 3
Election message is sent from 3 to 4
Ok message is sent from 4 to 3
Election message is sent from 4 to 5
Election message is sent from 3 to 5
Final coordinator is 4
Enter Your Choice
1. Crash Process
2. Recover Process
3. Exit
> _
```