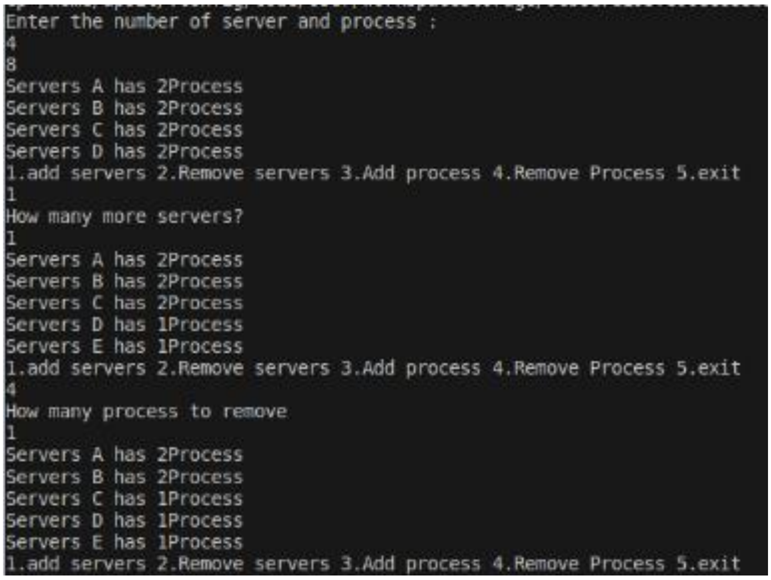
Load Balancing/Task Assign Usind static load Balance

```
import java.util.Scanner;

class Exp1 {
    static void printLoad(int servers, int Processes) {
        int each = Processes / servers;
        int extra = Processes % servers;
        int total = 0;
        for (int i = 0; i < servers; i++) {
            if (extra-- > 0)
                total = each + 1;
            else
                total = each;
            System.out.println("Server " + (char) ('A' + i) + " has " + total + " processes");
        }
    }

    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of servers and processes:");
        int servers = sc.nextInt();
        int Processes = sc.nextInt();
        while (true) {
            printLoad(servers, Processes);
            System.out.println("1. Add servers 2. Remove servers 3. Add processes 4. Remove processes
5. Exit");
            switch (sc.nextInt()) {
                case 1:
                    System.out.println("How many more servers?");
                    servers += sc.nextInt();
                    break;
                case 2:
                    System.out.println("How many servers to remove?");
                    servers -= sc.nextInt();
                    break;
                case 3:
                    System.out.println("How many more processes?");
                    Processes += sc.nextInt();
                    break;
                case 4:
                    System.out.println("How many processes to remove?");
                    Processes -= sc.nextInt();
                    break;
                case 5:
                    return;
            }
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Exp1.main(args); // Call the main method of the Exp1 class
    }
}
```

Run: Java Exp1.java



```
Enter the number of server and process :
4
8
Servers A has 2Process
Servers B has 2Process
Servers C has 2Process
Servers D has 2Process
1.add servers 2.Remove servers 3.Add process 4.Remove Process 5.exit
1
How many more servers?
1
Servers A has 2Process
Servers B has 2Process
Servers C has 2Process
Servers D has 1Process
Servers E has 1Process
1.add servers 2.Remove servers 3.Add process 4.Remove Process 5.exit
4
How many process to remove
1
Servers A has 2Process
Servers B has 2Process
Servers C has 1Process
Servers D has 1Process
Servers E has 1Process
1.add servers 2.Remove servers 3.Add process 4.Remove Process 5.exit
```

Process Commuincation Using RPC

Server.java

```
import java.io.*;
import java.net.*;
class server {
public static void main(String[] args) throws Exception{
ServerSocket sersock = new ServerSocket(3000);
System.out.println("Server ready");
Socket sock = sersock.accept( );
BufferedReader keyRead = new BufferedReader(new InputStreamReader(System.in));
OutputStream ostream = sock.getOutputStream();
PrintWriter pwrite = new PrintWriter(ostream, true);
InputStream istream = sock.getInputStream();
BufferedReader receiveRead = new BufferedReader(new
InputStreamReader(istream));
String receiveMessage, sendMessage,fun;
int a,b,c;
while(true)
{
fun = receiveRead.readLine();
if(fun != null)
 System.out.println("Operation : "+fun);
a = Integer.parseInt(receiveRead.readLine());
System.out.println("Parameter 1 : "+a);
b = Integer.parseInt(receiveRead.readLine());
if(fun.compareTo("add") == 0){

 c=a+b;
 System.out.println("Addition = "+c);
 pwrite.println("Addition = "+c);
}
 if(fun.compareTo("sub") == 0){
 c=a-b;
 System.out.println("Substraction = "+c);
 pwrite.println("Substraction = "+c);
 }
if(fun.compareTo("mul") == 0)
{
c=a*b;
System.out.println("Multiplication ="+c);
pwrite.println("Multiplication ="+c);
if(fun.compareTo("div") == 0)
{
c=a/b;
System.out.println("Division = "+c);
pwrite.println("Division = "+c);
System.out.flush();
}
}
}
}
}
```

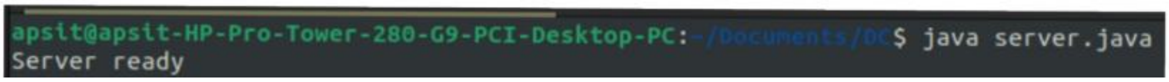
Client.java

```
import java.io.*;
import java.net.*;
class client{
public static void main(String[] args) throws Exception
 {
 Socket sock = new Socket("127.0.0.1", 3000);
 BufferedReader keyRead = new BufferedReader(new
InputStreamReader(System.in));
 OutputStream ostream = sock.getOutputStream();
 PrintWriter pwrite = new PrintWriter(ostream, true);
 InputStream istream = sock.getInputStream();
 BufferedReader receiveRead = new BufferedReader(new
InputStreamReader(istream));
 System.out.println("Client ready, type and press Enter key");
 String receiveMessage, sendMessage,temp;
 while(true)
 {
 System.out.println("InEnter operation to perform(add,sub,mul,div) .... ");
 temp = keyRead.readLine();
 sendMessage=temp.toLowerCase();
 pwrite.println(sendMessage);
 System.out.println("Enter first parameter :");
 sendMessage = keyRead.readLine();
 pwrite.println(sendMessage);
 System.out.println("Enter second parameter : ");
 sendMessage = keyRead.readLine();
 pwrite.println(sendMessage);
 System.out.flush();
 if((receiveMessage = receiveRead.readLine()) != null)
 System.out.println(receiveMessage);

 }
}
}
```

Run : Java Server.java (sirf ye run krke ruk jao fhir dusra terminal open kro0

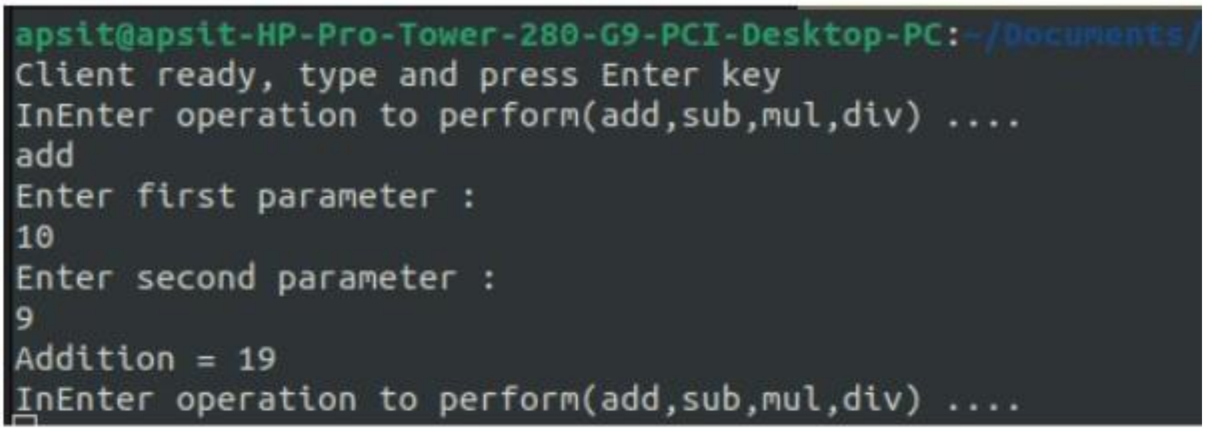
Run : Java Client.java(perform operations)

Server



```
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Desktop-PC:~/Documents/DC$ java server.java
Server ready
```

Client :



```
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Desktop-PC:~/Documents/
Client ready, type and press Enter key
InEnter operation to perform(add,sub,mul,div) ....
add
Enter first parameter :
10
Enter second parameter :
9
Addition = 19
InEnter operation to perform(add,sub,mul,div) ....
```

Inter Process Communication using Java

IPCServer.java

```
import java.net.*;
import java.io.*;
public class IPCServer {
 public static void main(String[] args) {
 System.out.println("INTERPROCESS COMMUNICATION SERVER PROCESS STARTED");
 System.out.println("SERVER IS READY AND WAITING TO RECEIVE DATA FROM CLIENT
PROCESS ON PORT 1200");
 try {
 ServerSocket ss = new ServerSocket(1200); // 1200 is the port number
 Socket clientSocket = ss.accept();
 System.out.println("\nClient is connected with IP address " +
clientSocket.getInetAddress()
 + " and port Number " + clientSocket.getPort());
 DataOutputStream dos = new
DataOutputStream(clientSocket.getOutputStream());
 DataInputStream dis = new
DataInputStream(clientSocket.getInputStream());
 int a = dis.readInt();
 System.out.println("\nNumber 1 ---- " + a);
 int b = dis.readInt();
 System.out.println("\nNumber 2 ---- " + b);
 int c = a + b;
 dos.writeInt(c);
 System.out.println("\nSERVER PROCESS HAS EXECUTED REQUESTED PROCESS AND SENT
RESULT " + c + " TO THE CLIENT");
 clientSocket.close();
 System.out.println("SERVER PROCESS EXITING.........");
 ss.close();
 } catch (Exception e) {
 e.printStackTrace();
 }
 }
}
```

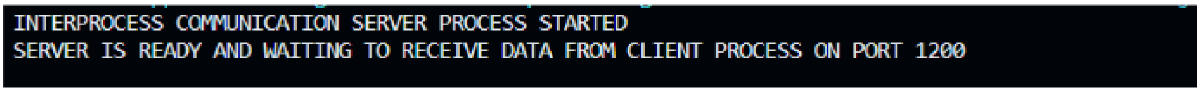
IPCClient.java

```
import java.net.*;
import java.io.*;
public class IPCClient {
 public static void main(String args[]) {
 try {
 Socket s = new Socket("localhost", 1200);
 DataOutputStream dos = new DataOutputStream(s.getOutputStream());
 DataInputStream dis = new DataInputStream(s.getInputStream());
 InputStreamReader isr = new InputStreamReader(System.in);
 System.out.println("\nCLIENT PROCESS STARTED");
 System.out.println("PLEASE ENTER THE VALUES OF Number 1 AND Number 2 TO PASS
THEM TO SERVER PROCESS");
 BufferedReader br = new BufferedReader(isr);
 System.out.print("Number 1: ");
 int a = Integer.parseInt(br.readLine());
 dos.writeInt(a);
 System.out.print("Number 2: ");
 int b = Integer.parseInt(br.readLine());
 dos.writeInt(b);
 int result = dis.readInt();
 System.out.println("\nCLIENT PROCESS HAS RECEIVED RESULT FROM SERVER");
 System.out.println("\nTHE ADDITION OF " + a + " AND " + b + " IS " + result);
 s.close();
 } catch (Exception e) {
 System.out.println("Exception is " + e);
 }
 }
}
```

Run: Java IPCServer.java(then wait and switch to 2nd terminal)

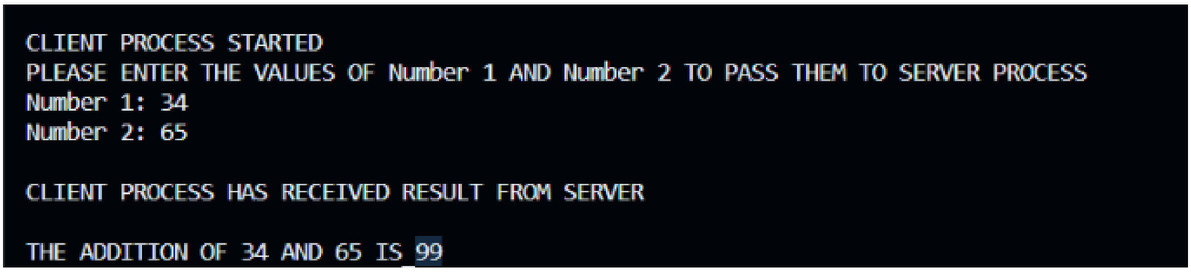
Run: Java IPCClient.java(give two nos)

Server



```
INTERPROCESS COMMUNICATION SERVER PROCESS STARTED
SERVER IS READY AND WAITING TO RECEIVE DATA FROM CLIENT PROCESS ON PORT 1200
```

Client



```
CLIENT PROCESS STARTED
PLEASE ENTER THE VALUES OF Number 1 AND Number 2 TO PASS THEM TO SERVER PROCESS
Number 1: 34
Number 2: 65

CLIENT PROCESS HAS RECEIVED RESULT FROM SERVER

THE ADDITION OF 34 AND 65 IS 99
```

Group Communication

```
//Server.java
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Vector;
```

```
public class Server {
private static Vector<PrintWriter> writers = new Vector<PrintWriter>();
public static void main(String[] args) throws Exception {
ServerSocket listener = new ServerSocket(9001);
System.out.println("The server is running at port 9001.");
while (true)
new Handler(listener.accept()).start();
}
private static class Handler extends Thread {
private Socket socket;
public Handler(Socket socket) {
this.socket = socket;
}
public void run() {
try {
BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
out.println("SUBMITNAME");
String name = in.readLine();
System.out.println(name + " joined");
writers.add(out);
while (true) {
String input = in.readLine();
for (PrintWriter writer : writers)
writer.println("MESSAGE" + name + ":" + input);
}
} catch (Exception e) {
System.err.println(e);
}
}
}
}
```

Master.java

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;
import java.util.Scanner;
public class master {
public static void main(String[] args) throws Exception {
Scanner sc = new Scanner(System.in);
Socket socket = new Socket("localhost", 9001);
BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
System.out.print("Enter your name: ");
String name = sc.nextLine();
while (true) {
String line = in.readLine();
if (line.startsWith("SUBMITNAME")) {
out.println(name);
} else if (line.startsWith("MESSAGE")) {
System.out.println(line.substring(8));
}
if(name.startsWith("master")) {
System.out.print("Enter a message: ");
out.println(sc.nextLine());
}
}
}
}
```

Slave1.java

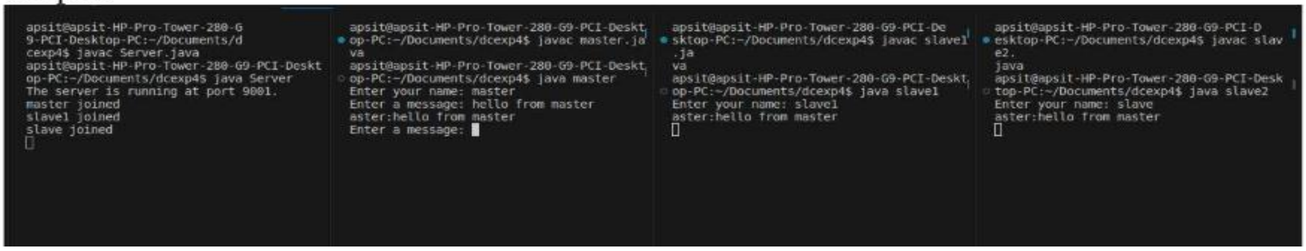
```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;
import java.util.Scanner;
public class slave1 {
public static void main(String[] args) throws Exception {
Scanner sc = new Scanner(System.in);
Socket socket = new Socket("localhost", 9001);
BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
System.out.print("Enter your name: ");
String name = sc.nextLine();
while (true) {
String line = in.readLine();
if (line.startsWith("SUBMITNAME")) {
out.println(name);
} else if (line.startsWith("MESSAGE")) {
System.out.println(line.substring(8));
}
if(name.startsWith("master")) {
System.out.print("Enter a message: ");
out.println(sc.nextLine());
}
}
}
}
```

Slave2.java

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;
import java.util.Scanner;
public class slave2 {
public static void main(String[] args) throws Exception {
Scanner sc = new Scanner(System.in);
Socket socket = new Socket("localhost", 9001);
BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
System.out.print("Enter your name: ");
String name = sc.nextLine();
while (true) {
String line = in.readLine();
if (line.startsWith("SUBMITNAME")) {
out.println(name);
} else if (line.startsWith("MESSAGE")) {
System.out.println(line.substring(8));
}
if(name.startsWith("master")) {
System.out.print("Enter a message: ");
out.println(sc.nextLine());
}
}
}
}
```

Run : java Server.java
java master.java, Java slave1.java, java slave2.java

output



```
apsit@apsit-HP-Pro-Tower-280-G
9-PCI-Desktop-PC:~/Documents/d
cexp4$ javac Server.java
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Deskt
op-PC:~/Documents/dcexp4$ java Server
The server is running at port 9001.
master joined
slave1 joined
slave joined
▯
```

```
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Deskt
op-PC:~/Documents/dcexp4$ javac master.ja
va
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Deskt
op-PC:~/Documents/dcexp4$ java master
Enter your name: master
Enter a message: hello from master
aster:hello from master
Enter a message: █
```

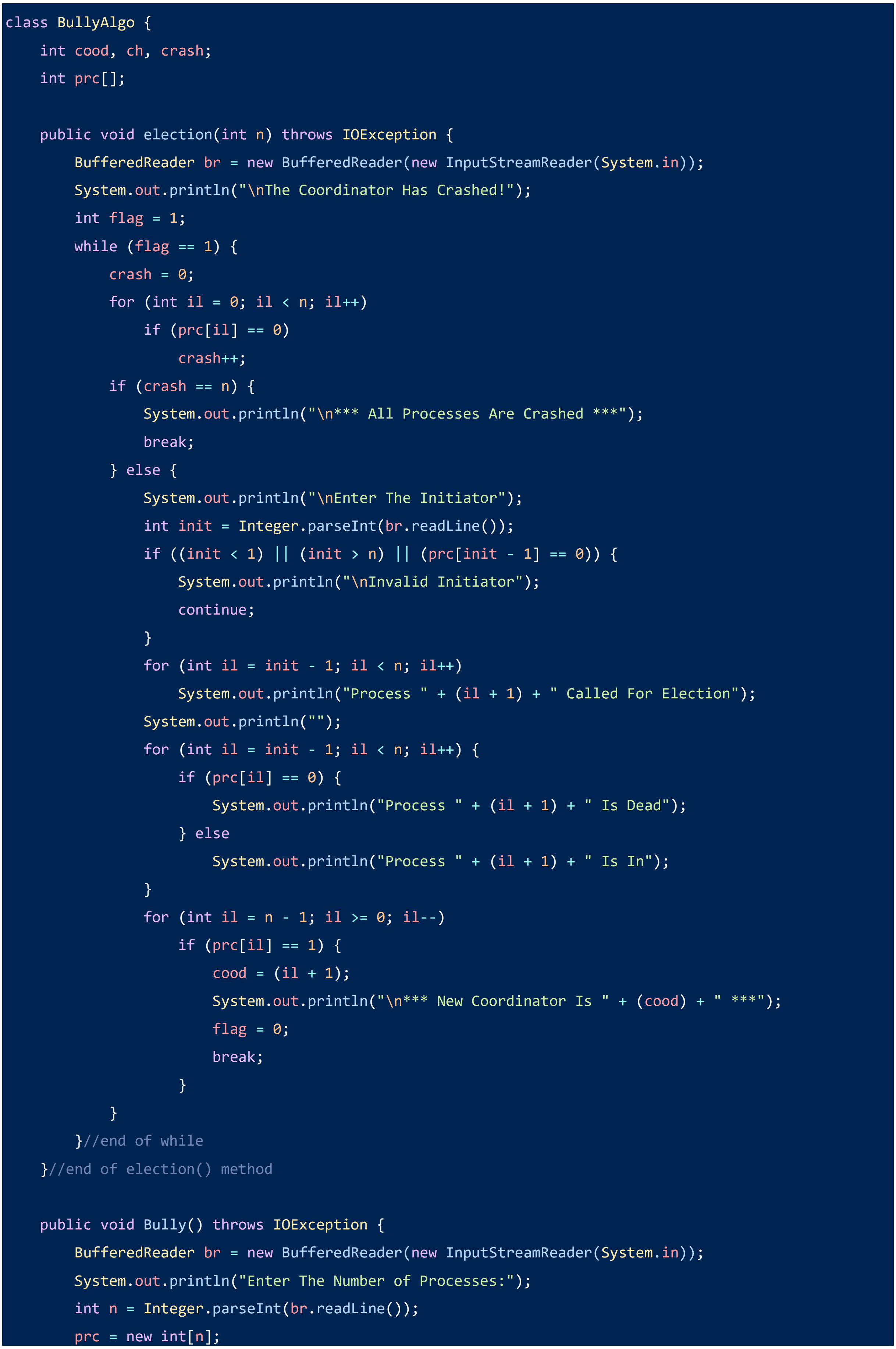
```
apsit@apsit-HP-Pro-Tower-280-G9-PCI-De
sktop-PC:~/Documents/dcexp4$ javac slave1
.ja
va
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Deskt
op-PC:~/Documents/dcexp4$ java slave1
Enter your name: slave1
aster:hello from master
▯
```

```
apsit@apsit-HP-Pro-Tower-280-G9-PCI-D
esktop-PC:~/Documents/dcexp4$ javac slav
e2.
java
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Desk
top-PC:~/Documents/dcexp4$ java slave2
Enter your name: slave
aster:hello from master
▯
```

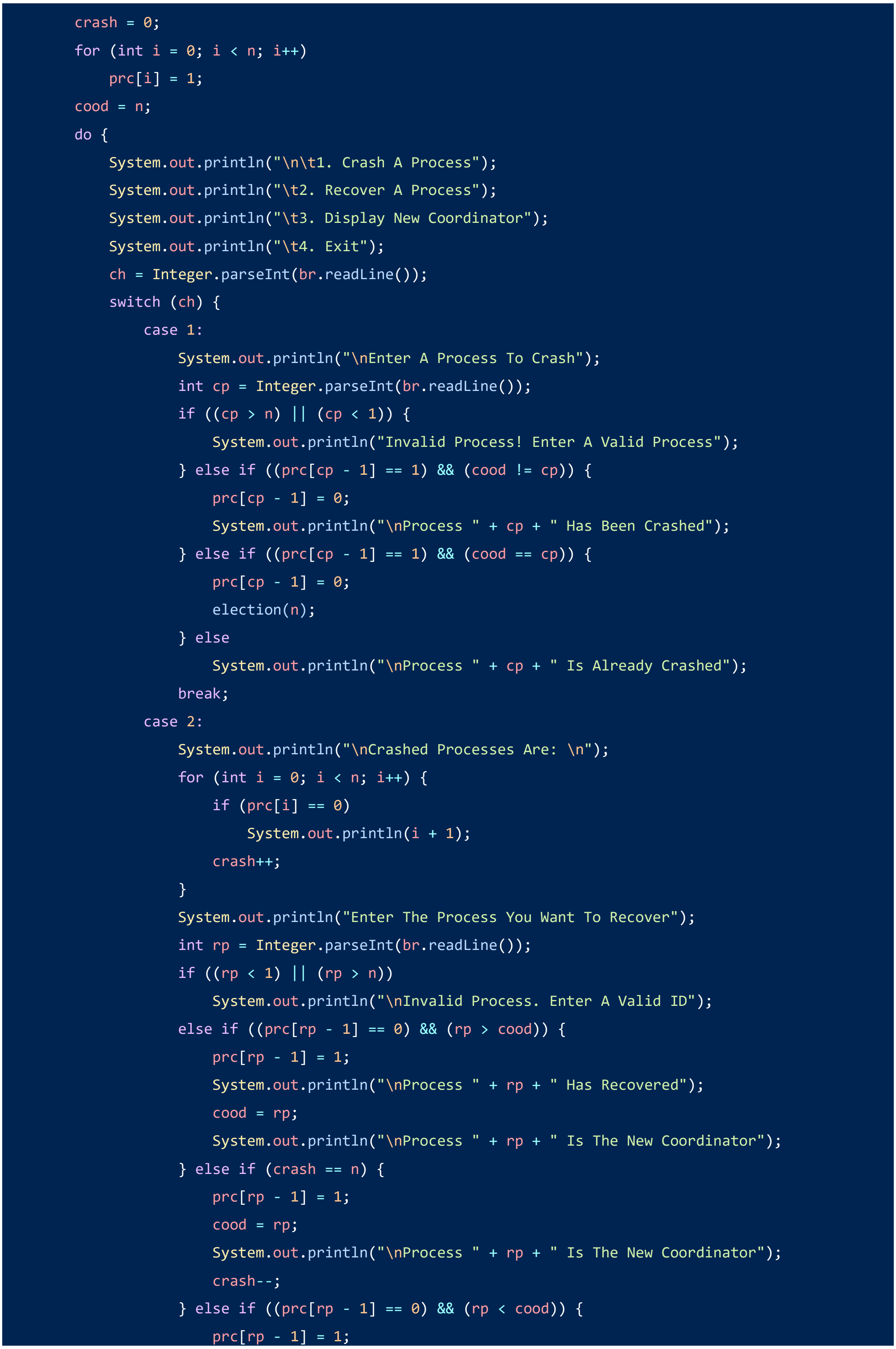
(combined output)

BullyAlgo:

```
import java.io.*;
```

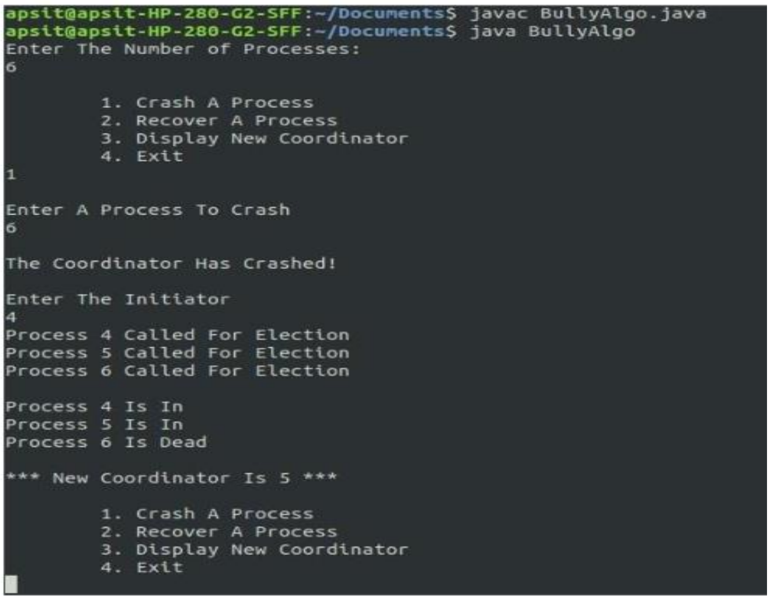


```
class BullyAlgo {
    int cood, ch, crash;
    int prc[];

    public void election(int n) throws IOException {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("\nThe Coordinator Has Crashed!");
        int flag = 1;
        while (flag == 1) {
            crash = 0;
            for (int il = 0; il < n; il++)
                if (prc[il] == 0)
                    crash++;
            if (crash == n) {
                System.out.println("\n*** All Processes Are Crashed ***");
                break;
            } else {
                System.out.println("\nEnter The Initiator");
                int init = Integer.parseInt(br.readLine());
                if ((init < 1) || (init > n) || (prc[init - 1] == 0)) {
                    System.out.println("\nInvalid Initiator");
                    continue;
                }
                for (int il = init - 1; il < n; il++)
                    System.out.println("Process " + (il + 1) + " Called For Election");
                System.out.println("");
                for (int il = init - 1; il < n; il++) {
                    if (prc[il] == 0) {
                        System.out.println("Process " + (il + 1) + " Is Dead");
                    } else
                        System.out.println("Process " + (il + 1) + " Is In");
                }
                for (int il = n - 1; il >= 0; il--)
                    if (prc[il] == 1) {
                        cood = (il + 1);
                        System.out.println("\n*** New Coordinator Is " + (cood) + " ***");
                        flag = 0;
                        break;
                    }
            }
        }//end of while
    }//end of election() method

    public void Bully() throws IOException {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter The Number of Processes:");
        int n = Integer.parseInt(br.readLine());
        prc = new int[n];
```



```java
        crash = 0;
        for (int i = 0; i < n; i++)
            prc[i] = 1;
        cood = n;
        do {
            System.out.println("\n\t1. Crash A Process");
            System.out.println("\t2. Recover A Process");
            System.out.println("\t3. Display New Coordinator");
            System.out.println("\t4. Exit");
            ch = Integer.parseInt(br.readLine());
            switch (ch) {
                case 1:
                    System.out.println("\nEnter A Process To Crash");
                    int cp = Integer.parseInt(br.readLine());
                    if ((cp > n) || (cp < 1)) {
                        System.out.println("Invalid Process! Enter A Valid Process");
                    } else if ((prc[cp - 1] == 1) && (cood != cp)) {
                        prc[cp - 1] = 0;
                        System.out.println("\nProcess " + cp + " Has Been Crashed");
                    } else if ((prc[cp - 1] == 1) && (cood == cp)) {
                        prc[cp - 1] = 0;
                        election(n);
                    } else
                        System.out.println("\nProcess " + cp + " Is Already Crashed");
                    break;
                case 2:
                    System.out.println("\nCrashed Processes Are: \n");
                    for (int i = 0; i < n; i++) {
                        if (prc[i] == 0)
                            System.out.println(i + 1);
                        crash++;
                    }
                    System.out.println("Enter The Process You Want To Recover");
                    int rp = Integer.parseInt(br.readLine());
                    if ((rp < 1) || (rp > n))
                        System.out.println("\nInvalid Process. Enter A Valid ID");
                    else if ((prc[rp - 1] == 0) && (rp > cood)) {
                        prc[rp - 1] = 1;
                        System.out.println("\nProcess " + rp + " Has Recovered");
                        cood = rp;
                        System.out.println("\nProcess " + rp + " Is The New Coordinator");
                    } else if (crash == n) {
                        prc[rp - 1] = 1;
                        cood = rp;
                        System.out.println("\nProcess " + rp + " Is The New Coordinator");
                        crash--;
                    } else if ((prc[rp - 1] == 0) && (rp < cood)) {
                        prc[rp - 1] = 1;
                        System.out.println("\nProcess " + rp + " Has Recovered");
                    } else
                        System.out.println("\nProcess " + rp + " Is Not A Crashed Process");
                    break;
                case 3:
                    System.out.println("\nCurrent Coordinator Is " + cood);
                    break;
                case 4:
                    System.exit(0);
                    break;
                default:
                    System.out.println("\nInvalid Entry!");
                    break;
            }//end switch
        } while (ch != 4);
    }//end of Bully()

    public static void main(String args[]) throws IOException {
        BullyAlgo ob = new BullyAlgo();
        ob.Bully();
    }
}
```

Run: java BullyAlgo.java



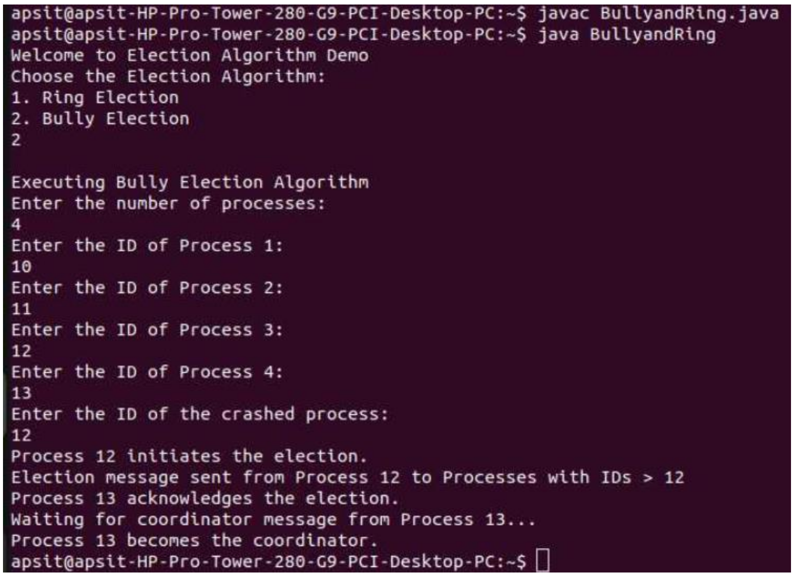
```
apsit@apsit-HP-280-G2-SFF:~/Documents$ javac BullyAlgo.java
apsit@apsit-HP-280-G2-SFF:~/Documents$ java BullyAlgo
Enter The Number of Processes:
6

        1. Crash A Process
        2. Recover A Process
        3. Display New Coordinator
        4. Exit
1

Enter A Process To Crash
6

The Coordinator Has Crashed!

Enter The Initiator
4
Process 4 Called For Election
Process 5 Called For Election
Process 6 Called For Election

Process 4 Is In
Process 5 Is In
Process 6 Is Dead

*** New Coordinator Is 5 ***

        1. Crash A Process
        2. Recover A Process
        3. Display New Coordinator
        4. Exit
```

Election Algo(Bully and Ring)

```
import java.io.*;
public class BullyandRing {
 public static void main(String[] args) throws IOException {
 BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
 System.out.println("Welcome to Election Algorithm Demo");
 System.out.println("Choose the Election Algorithm:");
 System.out.println("1. Ring Election");
 System.out.println("2. Bully Election");
 int choice = Integer.parseInt(br.readLine());
 switch (choice) {
 case 1:
 System.out.println("\nExecuting Ring Election Algorithm");
 RingElection ring = new RingElection();
 ring.execute();
 break;
 case 2:
 System.out.println("\nExecuting Bully Election Algorithm");
 BullyElection bully = new BullyElection();
 bully.execute();
 break;
 default:
 System.out.println("Invalid choice. Exiting...");
 }
 }
}
class RingElection {
 public void execute() {
 try {
 BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
 System.out.println("Enter the number of processes:");
 int n = Integer.parseInt(br.readLine());
 int[] processes = new int[n];
 for (int i = 0; i < n; i++) {
 System.out.println("Enter the ID of Process " + (i + 1) + ":");
 processes[i] = Integer.parseInt(br.readLine());
 }
 System.out.println("Enter the initiator process ID:");
 int initiator = Integer.parseInt(br.readLine());
 // Find the next process in the ring
 int nextProcess = -1;
 for (int i = 0; i < n; i++) {
 if (processes[i] == initiator) {
 nextProcess = processes[(i + 1) % n];
 break;
 }
 }
 // Broadcast election message
 System.out.println("Process " + initiator + " initiates the election.");
 System.out.println("Election message passed from Process " + initiator + " to
Process " + nextProcess);
 // Simulate election result
 int coordinator = processes[n - 1]; // Last process becomes coordinator
 System.out.println("Process " + coordinator + " becomes the coordinator.");
 } catch (IOException e) {
```

```
 e.printStackTrace();
 }
 }
}
class BullyElection {
 public void execute() {
 try {
 BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
 System.out.println("Enter the number of processes:");
 int n = Integer.parseInt(br.readLine());
 int[] processes = new int[n];
 for (int i = 0; i < n; i++) {
 System.out.println("Enter the ID of Process " + (i + 1) + ":");
 processes[i] = Integer.parseInt(br.readLine());
 }
 System.out.println("Enter the ID of the crashed process:");
 int crashedProcess = Integer.parseInt(br.readLine());
 // Election procedure
 int highestProcessId = 0;
 for (int process : processes) {
 if (process > highestProcessId && process != crashedProcess) {
    highestProcessId = process;
 }
 }
 if (highestProcessId == 0) {
 // Current process is the highest, become coordinator
 System.out.println("Process " + crashedProcess + " becomes the
coordinator.");
 } else {
 // Send election message to processes with higher IDs
 System.out.println("Process " + crashedProcess + " initiates the election.");
 System.out.println("Election message sent from Process " + crashedProcess + "
to Processes with IDs > " + crashedProcess);
 // Simulate receiving OK messages from processes with higher IDs
 System.out.println("Process " + highestProcessId + " acknowledges the
election.");
 // Wait for coordinator message from the highest ID process
 System.out.println("Waiting for coordinator message from Process " +
highestProcessId +"...");
 System.out.println("Process " + highestProcessId + " becomes the
coordinator.");
 }
 } catch (IOException e) {
 e.printStackTrace();
 }
 }
}
```

Run: java BullyandRing.java



```
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Desktop-PC:~$ javac BullyandRing.java
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Desktop-PC:~$ java BullyandRing
Welcome to Election Algorithm Demo
Choose the Election Algorithm:
1. Ring Election
2. Bully Election
2

Executing Bully Election Algorithm
Enter the number of processes:
4
Enter the ID of Process 1:
10
Enter the ID of Process 2:
11
Enter the ID of Process 3:
12
Enter the ID of Process 4:
13
Enter the ID of the crashed process:
12
Process 12 initiates the election.
Election message sent from Process 12 to Processes with IDs > 12
Process 13 acknowledges the election.
Waiting for coordinator message from Process 13...
Process 13 becomes the coordinator.
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Desktop-PC:~$ 
```

Mutual Exclusion:

MutualServer.java

```java
import java.io.*;
import java.net.*;
public class MutualServer implements Runnable {
public Socket socket;
public static ServerSocket ss;
MutualServer(Socket newSocket) {
this.socket = newSocket;
}
public static void main(String[] args) {
try
{
ss = new ServerSocket(7000);
System.out.println("Server Started");
while (true)
{
Socket s = ss.accept();
MutualServer es = new MutualServer(s);
Thread t = new Thread(es);
t.start();
}
}catch (Exception e)
{
e.printStackTrace();
}
}
public void run()
{
try
{
BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
while (true)
{
System.out.println(in.readLine());
}
}catch (Exception e)
{
e.printStackTrace();
}
}
}
```

ClientOne.java

```
import java.io.*;
import java.net.*;
public class ClientOne
{
public static void main(String args[])throws IOException
{
Socket s= new Socket("localhost", 7000) ;
PrintStream out = new PrintStream(s.getOutputStream());
ServerSocket ss= new ServerSocket( 7001) ;
Socket s1= ss.accept();
BufferedReader inl = new BufferedReader(new
InputStreamReader(s1.getInputStream()));
PrintStream out1 = new PrintStream(s1.getOutputStream());
BufferedReader br= new BufferedReader(new InputStreamReader(System.in));
String str="Token";
while(true)
{
if(str.equalsIgnoreCase("Token"))
{
System.out.println("Do you want to send some data: ");
System.out.println("Enter Yes or no");
str=br.readLine();
if(str.equalsIgnoreCase("Yes"))
{
System.out.println("Enter the data");
str=br.readLine();
out.println(str);
}
out1.println("Token");
}
System.out.println("Waiting for Token");
str=inl.readLine();
}
}
}
```

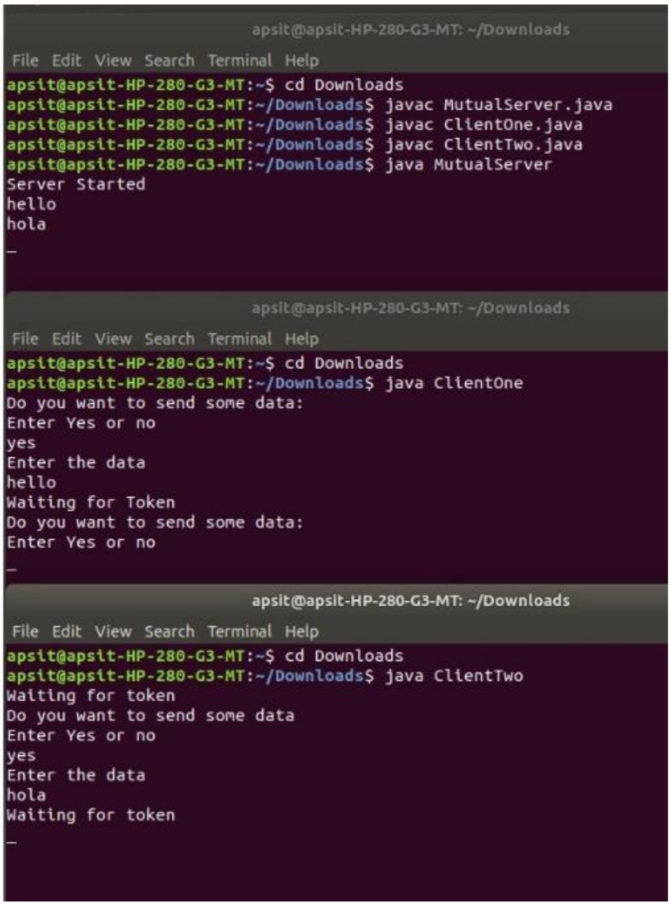
ClientTwo.java

```
import java.io.*;
import java.net.*;
public class ClientTwo
{
public static void main(String args[])throws IOException
{
Socket s= new Socket("localhost", 7000);
PrintStream out = new PrintStream(s.getOutputStream());
Socket s2= new Socket("localhost",7001);
BufferedReader in2= new BufferedReader(new
InputStreamReader(s2.getInputStream()));
PrintStream out2 = new PrintStream(s2.getOutputStream());
BufferedReader br= new BufferedReader(new InputStreamReader(System.in));
String str;
while(true)
{
System.out.println("Waiting for token");
str=in2.readLine();
if(str.equalsIgnoreCase("Token"))
{
System.out.println("Do you want to send some data");
System.out.println("Enter Yes or no");
str=br.readLine();
if(str.equalsIgnoreCase("Yes"))
{
System.out.println("Enter the data");
str=br.readLine();
out.println(str);
}
out2.println("Token");
}
}
}
}
```

Run: Java MutualServer.java(then wait)

Java ClientOne.java(wait)

Java ClientTwo.java(strt communication)



apsit@apsit-HP-280-G3-MT: ~/Downloads
File Edit View Search Terminal Help

```
apsit@apsit-HP-280-G3-MT:~$ cd Downloads
apsit@apsit-HP-280-G3-MT:~/Downloads$ javac MutualServer.java
apsit@apsit-HP-280-G3-MT:~/Downloads$ javac ClientOne.java
apsit@apsit-HP-280-G3-MT:~/Downloads$ javac ClientTwo.java
apsit@apsit-HP-280-G3-MT:~/Downloads$ java MutualServer
Server Started
hello
hola
_
```

apsit@apsit-HP-280-G3-MT: ~/Downloads
File Edit View Search Terminal Help

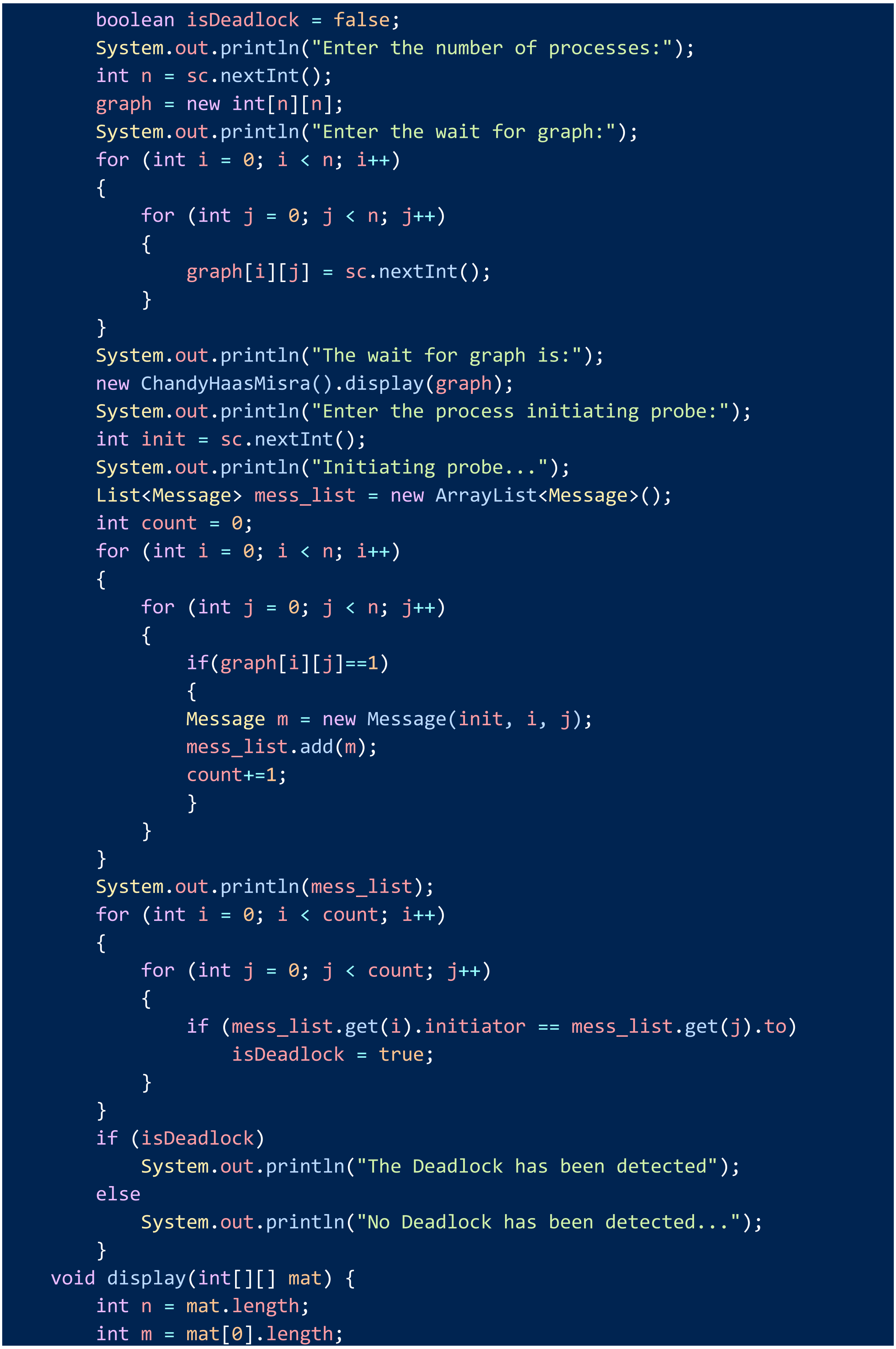
```
apsit@apsit-HP-280-G3-MT:~$ cd Downloads
apsit@apsit-HP-280-G3-MT:~/Downloads$ java ClientOne
Do you want to send some data:
Enter Yes or no
yes
Enter the data
hello
Waiting for Token
Do you want to send some data:
Enter Yes or no
_
```

apsit@apsit-HP-280-G3-MT: ~/Downloads
File Edit View Search Terminal Help

```
apsit@apsit-HP-280-G3-MT:~$ cd Downloads
apsit@apsit-HP-280-G3-MT:~/Downloads$ java ClientTwo
Waiting for token
Do you want to send some data
Enter Yes or no
yes
Enter the data
hola
Waiting for token
_
```

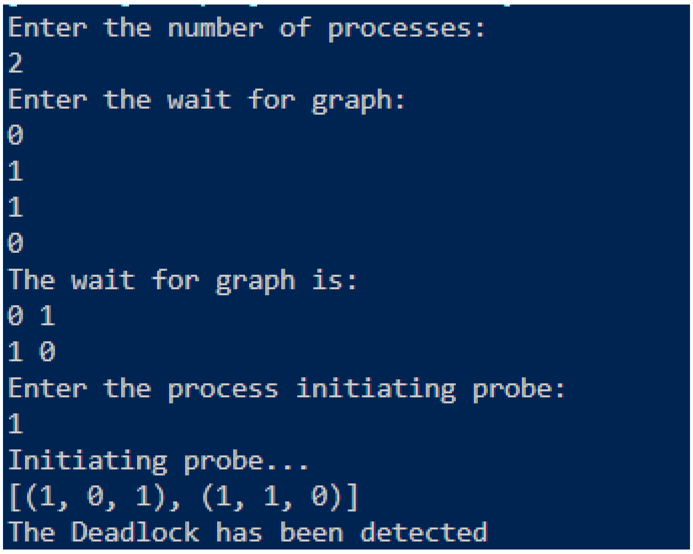
Chandy Mishra Distributed Deadlock Detection Algo:

```
import java.util.*;

class Message {
    public int initiator=0;
    public int from=0;
    public int to=0;
    public Message(int i, int j, int k)
    {
        initiator = i;
        from = j;
        to = k;
    }
    public String toString()
    {
        return "(" + initiator + ", " + from + ", " + to + ")";
    }
}
public class ChandyHaasMisra {
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int graph[][];
```



```
        boolean isDeadlock = false;
        System.out.println("Enter the number of processes:");
        int n = sc.nextInt();
        graph = new int[n][n];
        System.out.println("Enter the wait for graph:");
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                graph[i][j] = sc.nextInt();
            }
        }
        System.out.println("The wait for graph is:");
        new ChandyHaasMisra().display(graph);
        System.out.println("Enter the process initiating probe:");
        int init = sc.nextInt();
        System.out.println("Initiating probe...");
        List<Message> mess_list = new ArrayList<Message>();
        int count = 0;
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                if(graph[i][j]==1)
                {
                Message m = new Message(init, i, j);
                mess_list.add(m);
                count+=1;
                }
            }
        }
        System.out.println(mess_list);
        for (int i = 0; i < count; i++)
        {
            for (int j = 0; j < count; j++)
            {
                if (mess_list.get(i).initiator == mess_list.get(j).to)
                    isDeadlock = true;
            }
        }
        if (isDeadlock)
            System.out.println("The Deadlock has been detected");
        else
            System.out.println("No Deadlock has been detected...");
        }
    void display(int[][] mat) {
        int n = mat.length;
        int m = mat[0].length;
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++)
                System.out.print(mat[i][j] + " ");
            System.out.println();
        }
    }
}
```

Run: Java ChandyHaasMisra.java

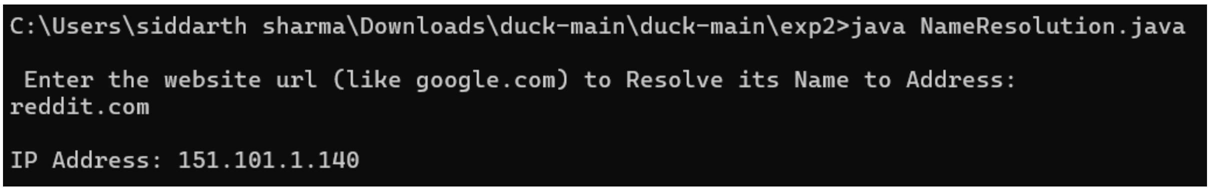


```
Enter the number of processes:
2
Enter the wait for graph:
0
1
1
0
The wait for graph is:
0 1
1 0
Enter the process initiating probe:
1
Initiating probe...
[(1, 0, 1), (1, 1, 0)]
The Deadlock has been detected
```

Demonstrate Name resolution using Java:

```
import java.io.*;
import java.net.*;
public class NameResolution
{
public static void main(String args[]) throws IOException
{
BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
System.out.println("\n Enter the website url (like google.com) to Resolve its
Name to Address: ");
String name = br.readLine();
try
{
InetAddress ip = InetAddress.getByName(name);
System.out.println("\nIP Address: "+ip.getHostAddress());
}
catch(UnknownHostException e)
{
System.out.println("\n\n No such Host is present...");
System.out.println("\n Try Again...");
}
}
}
```

Run: Java NameResolution.java



```
C:\Users\siddarth sharma\Downloads\duck-main\duck-main\exp2>java NameResolution.java

 Enter the website url (like google.com) to Resolve its Name to Address:
reddit.com

IP Address: 151.101.1.140
```