Assignment 3

Name : Omkar Deshpande

Roll No : 43212

Batch - Q10

Problem statement - To develop any distributed algorithm for leader election.

CODE -

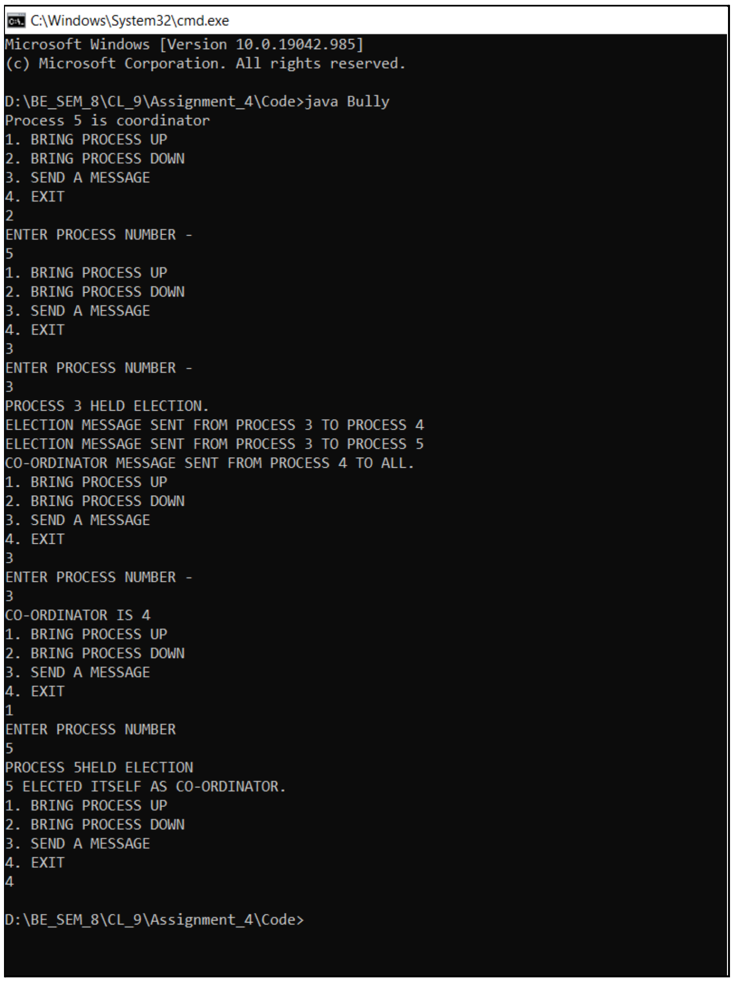
Bully.java

```
import java.io.InputStream;
import java.io.PrintStream;
import java.util.Scanner;

public class Bully
{
    static boolean[] state = new boolean[5];
    static int coordinator;

    public static void up(int up)//4
    {
        if (state[up - 1])// 0 1 2 3 4
        {
            System.out.println("PROCESS " + up + " IS ALREADY UP");
        }
        else
        {
            int i;
            Bully.state[up - 1] = true;
            System.out.println("PROCESS " + up + "HELD ELECTION");
            if (up > Bully.coordinator){
                System.out.println(up + " ELECTED ITSELF AS
CO-ORDINATOR.");
                Bully.coordinator = up;
                return;
            }
            for (i = up+1; i <= 5; i++)
            {
                System.out.println("ELECTION MESSAGE SENT FROM PROCESS "
+ up + "TO PROCESS " + i );
            }
            for (i = 5; i >= up; i--)
            {
                if (!state[i - 1]) continue;
                System.out.println("ALIVE MESSAGE SENT FROM PROCESS " +
i + "TO PROCESS" + up);
                break;
            }
        }
        System.out.println("PROCESS " + Bully.coordinator + " IS
CO-ORDINATOR.");
    }

    public static void down(int down)
    {
        if (!state[down - 1])
        {
            System.out.println("PROCESS " + down + " IS ALREADY DOWN.");
        }
        else
        {
            Bully.state[down - 1] = false;
        }
        if (Bully.coordinator==down){
            Bully.coordinator=Integer.MIN_VALUE;
        }
    }

    public static void mess(int mess)
    {
        if (state[mess - 1])
        {
            if (Bully.coordinator > mess)
            {
                System.out.println("CO-ORDINATOR IS " +
Bully.coordinator);
            }
            else
            {
                int i;
                System.out.println("PROCESS " + mess + " HELD 
ELECTION.");
                for (i = mess; i < 5; ++i)
                {
                    System.out.println("ELECTION MESSAGE SENT FROM 
PROCESS " + mess + " TO PROCESS " + (i + 1));
                }
                for (i = 5; i >= mess; i--)
                {
                    if (!state[i - 1]) continue;
                    System.out.println("CO-ORDINATOR MESSAGE SENT FROM 
PROCESS " + i + " TO ALL.");
                    Bully.coordinator=i;
                    break;
                }
            }
        }
        else
        {
            System.out.println("Prccess" + mess + "is down");
            Bully.up(mess);
        }
    }

    public static void main(String[] args)
    {
        int choice;
        Scanner sc = new Scanner(System.in);
        for (int i = 0; i < 5; ++i)
        {
            Bully.state[i] = true;
        }
        Bully.coordinator = 5;
        System.out.println("Process 5 is coordinator");
        do
        {
            System.out.println("1. BRING PROCESS UP");
            System.out.println("2. BRING PROCESS DOWN");
            System.out.println("3. SEND A MESSAGE");
            System.out.println("4. EXIT");
            choice = sc.nextInt();
            switch (choice)
            {
                case 1:
                {
                    System.out.println("ENTER PROCESS NUMBER");
                    int up = sc.nextInt();
                    if (up == Bully.coordinator)
                    {
                        System.out.println("Process " + up + " is
co-ordinator");
                        Bully.state[Bully.coordinator-1] = true;
                        break;
                    }
                    Bully.up(up);
                    break;
                }

                case 2:
                {
                    System.out.println("ENTER PROCESS NUMBER - ");
                    int down = sc.nextInt();
                    Bully.down(down);
                    break;
                }
                case 3:
                {
                    System.out.println("ENTER PROCESS NUMBER - ");
                    int mess = sc.nextInt();
                    Bully.mess(mess);
                }
            }
        } while (choice != 4);
    }
}
```

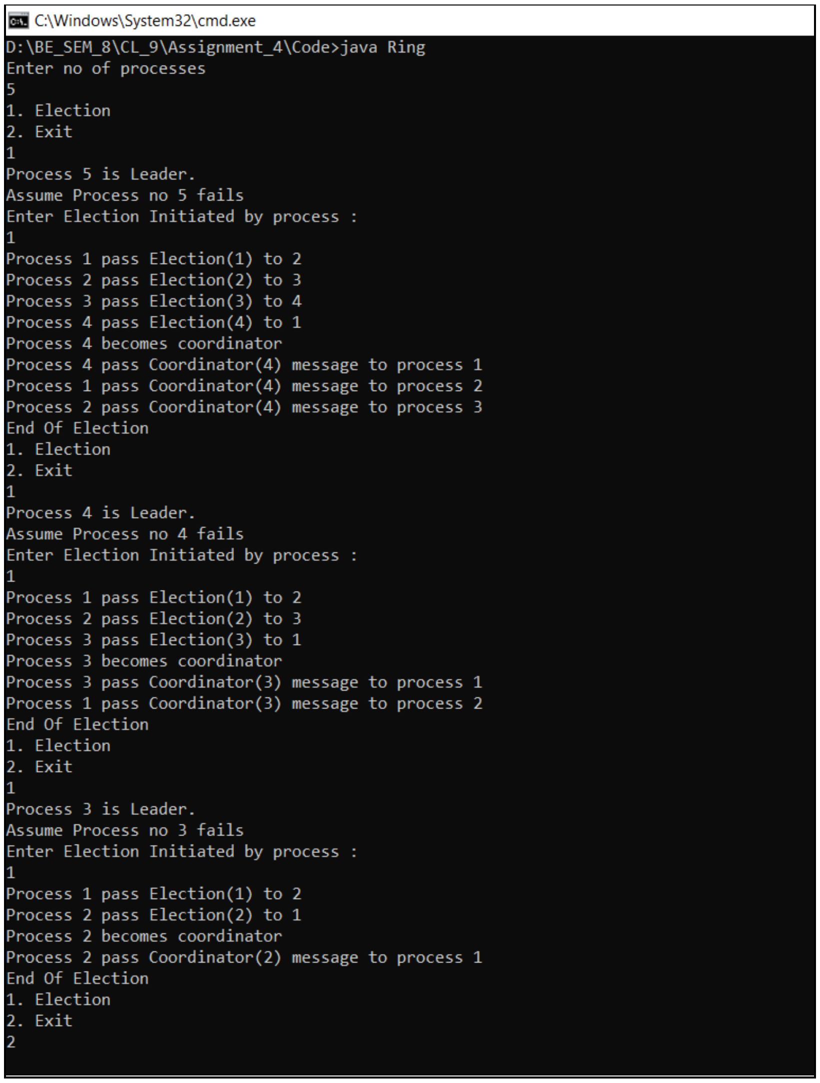
Ring.java

```java
import java.util.Scanner;

class Process{
    public int id;
    public boolean active;

    public Process(int id){
        this.id=id;
        active=true;
    }

    }
public class Ring{
    int noOfProcesses;
    Process[] processes;
    Scanner sc;

    public Ring(){
        sc=new Scanner(System.in);
    }
    public void initialiseRing(){
        System.out.println("Enter no of processes");
        noOfProcesses=sc.nextInt();
        processes = new Process[noOfProcesses];
        for(int i=0;i<processes.length;i++){
            processes[i]= new Process(i);
        }
    }

    public int getMax(){
        int maxId=-99;
        int maxIdIndex=0;
        for(int i=0;i<processes.length;i++){
            if(processes[i].active && processes[i].id>maxId){
                maxId=processes[i].id;
                maxIdIndex=i;
        }
        }
        return maxIdIndex;
    }

    public void performElection(){
        if (processes[getMax()].id == 0){
            System.out.println("No processes left.");
            return;
        }
        System.out.println("Process "+ (processes[getMax()].id + 1) + "
is Leader.");
        System.out.println("Assume Process no " +
(processes[getMax()].id + 1) +" fails");
        processes[getMax()].active=false;
        System.out.println("Enter Election Initiated by process : ");
        int initiatorProcesss=sc.nextInt();

        int prev = initiatorProcesss-1;
        int next = prev+1;

        while(true){
            if(processes[next].active){
            System.out.println("Process "+ (processes[prev].id + 1) +"
pass Election("+ (processes[prev].id + 1) +") to " + (processes[next].id
+ 1));
            prev=next;
            }

        next = (next+1)%noOfProcesses;
        if(next == initiatorProcesss-1){
            System.out.println("Process "+ (processes[prev].id + 1) +"
pass Election("+ (processes[prev].id + 1) +") to " + (processes[next].id
+ 1));

            break;
        }
        }
        System.out.println("Process "+ (processes[getMax()].id + 1) + "
becomes coordinator");
        int coordinator = processes[getMax()].id;

        prev = coordinator;
        next =(prev+1)%noOfProcesses;

        while(true){

            if(processes[next].active)
            {
            System.out.println("Process "+ (processes[prev].id + 1) +"
pass Coordinator("+ (coordinator + 1) + ") message to process "+
(processes[next].id + 1)  );
            prev = next;
            }
            next = (next+1) % noOfProcesses;
            if(next == coordinator)
            {
            System.out.println("End Of Election ");
            break;
            }
        }

    }

    public static void main(String arg[]){
        Scanner sc=new Scanner(System.in);
        Ring r= new Ring();
        r.initialiseRing();
        while(true){
        System.out.println("1. Election\n2. Exit");
        int choice = sc.nextInt();
        switch(choice){
        case(1):
            r.performElection();
            break;
        case(2):
            return;
        default:
            return;
        }
        }
    }


}
```

OUTPUT -



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19042.985]
(c) Microsoft Corporation. All rights reserved.

D:\BE_SEM_8\CL_9\Assignment_4\Code>java Bully
Process 5 is coordinator
1. BRING PROCESS UP
2. BRING PROCESS DOWN
3. SEND A MESSAGE
4. EXIT
2
ENTER PROCESS NUMBER -
5
1. BRING PROCESS UP
2. BRING PROCESS DOWN
3. SEND A MESSAGE
4. EXIT
3
ENTER PROCESS NUMBER -
3
PROCESS 3 HELD ELECTION.
ELECTION MESSAGE SENT FROM PROCESS 3 TO PROCESS 4
ELECTION MESSAGE SENT FROM PROCESS 3 TO PROCESS 5
CO-ORDINATOR MESSAGE SENT FROM PROCESS 4 TO ALL.
1. BRING PROCESS UP
2. BRING PROCESS DOWN
3. SEND A MESSAGE
4. EXIT
3
ENTER PROCESS NUMBER -
3
CO-ORDINATOR IS 4
1. BRING PROCESS UP
2. BRING PROCESS DOWN
3. SEND A MESSAGE
4. EXIT
1
ENTER PROCESS NUMBER
5
PROCESS 5HELD ELECTION
5 ELECTED ITSELF AS CO-ORDINATOR.
1. BRING PROCESS UP
2. BRING PROCESS DOWN
3. SEND A MESSAGE
4. EXIT
4

D:\BE_SEM_8\CL_9\Assignment_4\Code>
```


C:\Windows\System32\cmd.exe

```
D:\BE_SEM_8\CL_9\Assignment_4\Code>java Ring
Enter no of processes
5
1. Election
2. Exit
1
Process 5 is Leader.
Assume Process no 5 fails
Enter Election Initiated by process :
1
Process 1 pass Election(1) to 2
Process 2 pass Election(2) to 3
Process 3 pass Election(3) to 4
Process 4 pass Election(4) to 1
Process 4 becomes coordinator
Process 4 pass Coordinator(4) message to process 1
Process 1 pass Coordinator(4) message to process 2
Process 2 pass Coordinator(4) message to process 3
End Of Election
1. Election
2. Exit
1
Process 4 is Leader.
Assume Process no 4 fails
Enter Election Initiated by process :
1
Process 1 pass Election(1) to 2
Process 2 pass Election(2) to 3
Process 3 pass Election(3) to 1
Process 3 becomes coordinator
Process 3 pass Coordinator(3) message to process 1
Process 1 pass Coordinator(3) message to process 2
End Of Election
1. Election
2. Exit
1
Process 3 is Leader.
Assume Process no 3 fails
Enter Election Initiated by process :
1
Process 1 pass Election(1) to 2
Process 2 pass Election(2) to 1
Process 2 becomes coordinator
Process 2 pass Coordinator(2) message to process 1
End Of Election
1. Election
2. Exit
2
```