Name: Harsh Chaudhari Class: BE-10
Batch: P-10 Roll No: 43215

Assignment 6

Code:

Bully.java

```
import java.util.*;

public class Bully {
    int coordinator;
    int max_processes;
    boolean processes[];


    public Bully(int max) {
        max_processes = max;
        processes = new boolean[max_processes];
        coordinator = max;

        System.out.println("Creating processes..");
        for(int i = 0; i < max; i++) {
            processes[i] =  true;
            System.out.println("P"+ (i+1) + " created");
        }
        System.out.println("Process P" + coordinator + " is the coordinator");

    }
    void displayProcesses() {
        for(int i = 0; i < max_processes; i++) {
            if(processes[i]) {
                System.out.println("P" + (i+1) + " is up");
            } else {
                System.out.println("P" + (i+1) + " is down");
            }
        }
        System.out.println("Process P" + coordinator + " is the coordinator");
    }

    void upProcess(int process_id) {
        if(!processes[process_id - 1]) {
            processes[process_id - 1] = true;
            System.out.println("Process " + process_id + " is now up.");
        } else {
            System.out.println("Process " + process_id + " is already up.");
        }
    }

    void downProcess(int process_id) {
        if(!processes[process_id - 1]) {
            System.out.println("Process " + process_id + " is already down.");
        } else {
            processes[process_id - 1] = false;
            System.out.println("Process " + process_id + " is down.");
        }
    }

    void runElection(int process_id) {
        coordinator = process_id;
        boolean keepGoing = true;

        for(int i = process_id; i < max_processes && keepGoing; i++) {
            System.out.println("Election message sent from process " + process_id + " to process " +
(i+1));

            if(processes[i]) {
                keepGoing = false;
                runElection(i + 1);
            }
        }
    }

    public static void main(String args[]) {
        Bully bully = null;
        int max_processes = 0, process_id = 0;
        int choice = 0;
        Scanner sc = new Scanner(System.in);

        while(true) {
            System.out.println("Bully Algorithm");
            System.out.println("1. Create processes");
            System.out.println("2. Display processes");
            System.out.println("3. Up a process");
            System.out.println("4. Down a process");
            System.out.println("5. Run election algorithm");
            System.out.println("6. Exit Program");
            System.out.print("Enter your choice:- ");
            choice = sc.nextInt();

            switch(choice) {
                case 1:
                    System.out.print("Enter the number of processes:- ");
                    max_processes = sc.nextInt();
                    bully = new Bully(max_processes);
                    break;
                case 2:
                    bully.displayProcesses();
                    break;
                case 3:
                    System.out.print("Enter the process number to up:- ");
                    process_id = sc.nextInt();
                    bully.upProcess(process_id);
                    break;
                case 4:
                    System.out.print("Enter the process number to down:- ");
                    process_id = sc.nextInt();
                    bully.downProcess(process_id);
                    break;
                case 5:
                    System.out.print("Enter the process number which will perform election:- ");
                    process_id = sc.nextInt();
                    bully.runElection(process_id);
                    bully.displayProcesses();
                    break;
                case 6:
                    System.exit(0);
                    break;
                default:
                    System.out.println("Error in choice. Please try again.");
                    break;
            }
        }
    }
}
```

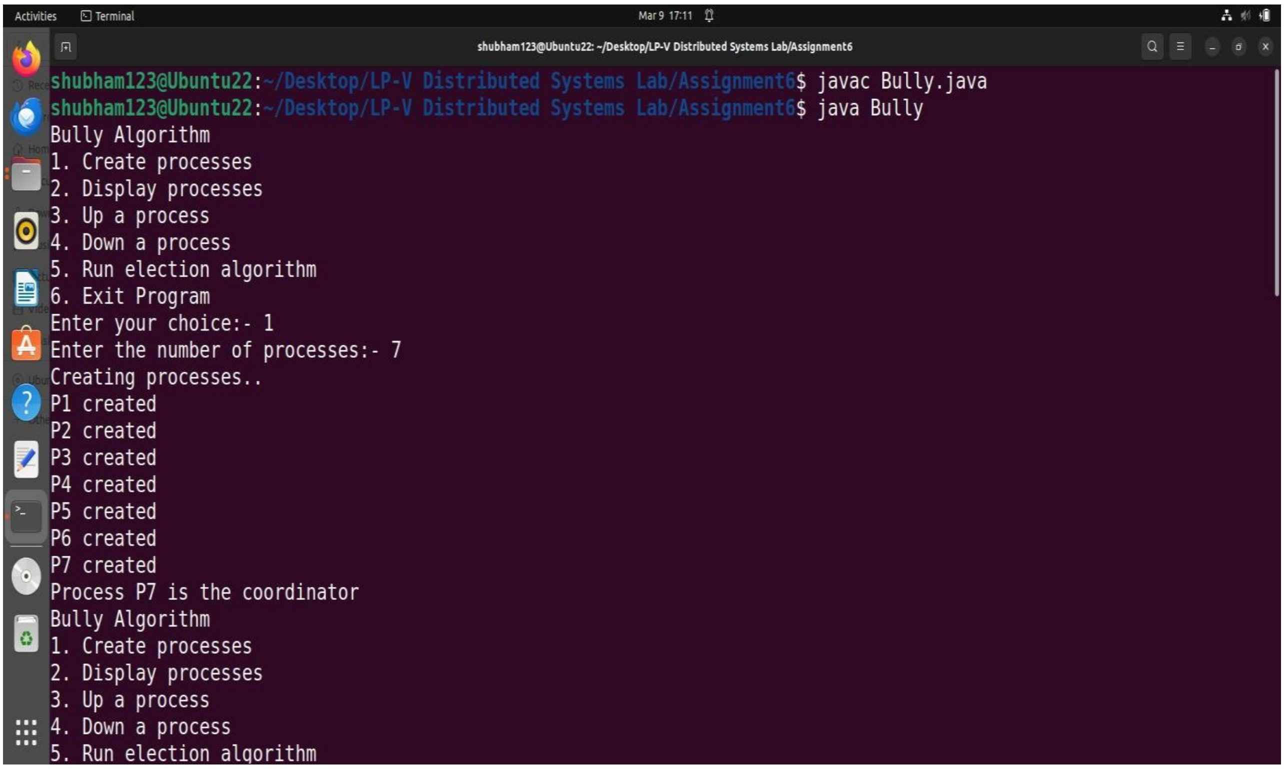
Ring.java

```
import java.util.*;

public class Ring {
    int max_processes;
    int coordinator;
    boolean processes[];
    ArrayList<Integer> pid;

    public Ring(int max) {
        coordinator = max;
        max_processes = max;
        pid = new ArrayList<Integer>();
        processes = new boolean[max];

        for(int i = 0; i < max; i++) {
            processes[i] = true;
            System.out.println("P" + (i+1) + " created.");
        }
        System.out.println("P" + (coordinator) + " is the coordinator");
    }

    void displayProcesses() {
        for(int i = 0; i < max_processes; i++) {
            if(processes[i])
                System.out.println("P" + (i+1) + " is up.");
            else
                System.out.println("P" + (i+1) + " is down.");
        }
        System.out.println("P" + (coordinator) + " is the coordinator");
    }

    void upProcess(int process_id) {
        if(!processes[process_id-1]) {
            processes[process_id-1] = true;
            System.out.println("Process P" + (process_id) + " is up.");
        } else {
            System.out.println("Process P" + (process_id) + " is already up.");
        }
    }

    void downProcess(int process_id) {
        if(!processes[process_id-1]) {
            System.out.println("Process P" + (process_id) + " is already down.");
        } else {
            processes[process_id-1] = false;
            System.out.println("Process P" + (process_id) + " is down.");
        }
    }

    void displayArrayList(ArrayList<Integer> pid) {
        System.out.print("[ ");
        for(Integer x : pid) {
            System.out.print(x + " ");
        }
        System.out.print(" ]\n");
    }

    void initElection(int process_id) {
        if(processes[process_id-1]) {
            pid.add(process_id);

            int temp = process_id;

            System.out.print("Process P" + process_id + " sending the following list:- ");
            displayArrayList(pid);

            while(temp != process_id - 1) {
                if(processes[temp]) {
                    pid.add(temp+1);
                    System.out.print("Process P" + (temp + 1) + " sending the following list:- ");
                    displayArrayList(pid);
                }
                temp = (temp + 1) % max_processes;
            }
            coordinator = Collections.max(pid);
            System.out.println("Process P" + process_id + " has declared P" + coordinator + " as the
 coordinator");
            pid.clear();
        }
    }
    public static void main(String args[]) {
        Ring ring = null;
        int max_processes = 0, process_id = 0;
        int choice = 0;
        Scanner sc = new Scanner(System.in);

        while(true) {
            System.out.println("Ring Algorithm");
            System.out.println("1. Create processes");
            System.out.println("2. Display processes");
            System.out.println("3. Up a process");
            System.out.println("4. Down a process");
            System.out.println("5. Run election algorithm");
            System.out.println("6. Exit Program");
            System.out.print("Enter your choice:- ");
            choice = sc.nextInt();

            switch(choice) {
                case 1:
                    System.out.print("Enter the total number of processes:- ");
                    max_processes = sc.nextInt();
                    ring = new Ring(max_processes);
                    break;
                case 2:
                    ring.displayProcesses();
                    break;
                case 3:
                    System.out.print("Enter the process to up:- ");
                    process_id = sc.nextInt();
                    ring.upProcess(process_id);
                    break;
                case 4:
                    System.out.print("Enter the process to down:- ");
                    process_id = sc.nextInt();
                    ring.downProcess(process_id);
                    break;
                case 5:
                    System.out.print("Enter the process which will initiate election:- ");
                    process_id = sc.nextInt();
                    ring.initElection(process_id);
                    break;
                case 6:
                    System.exit(0);
                    break;
                default:
                    System.out.println("Error in choice. Please try again.");
                    break;
            }
        }
    }
```

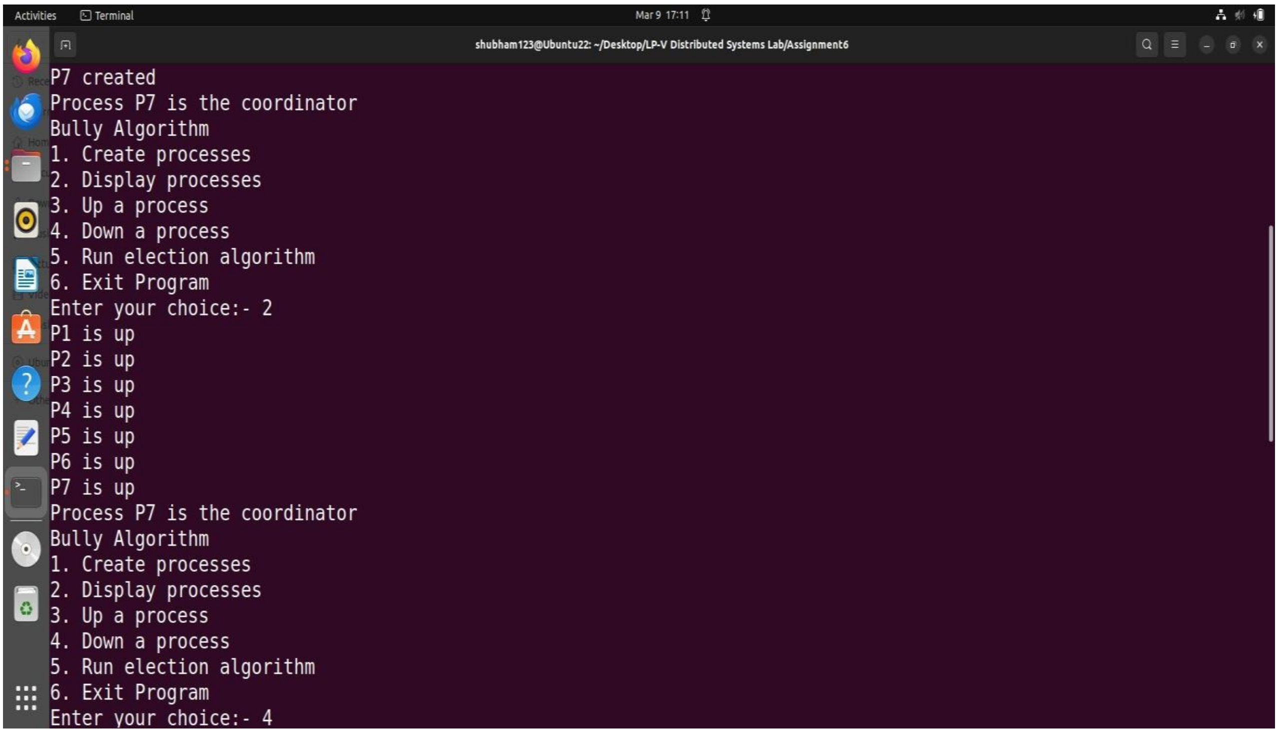
}

Output:

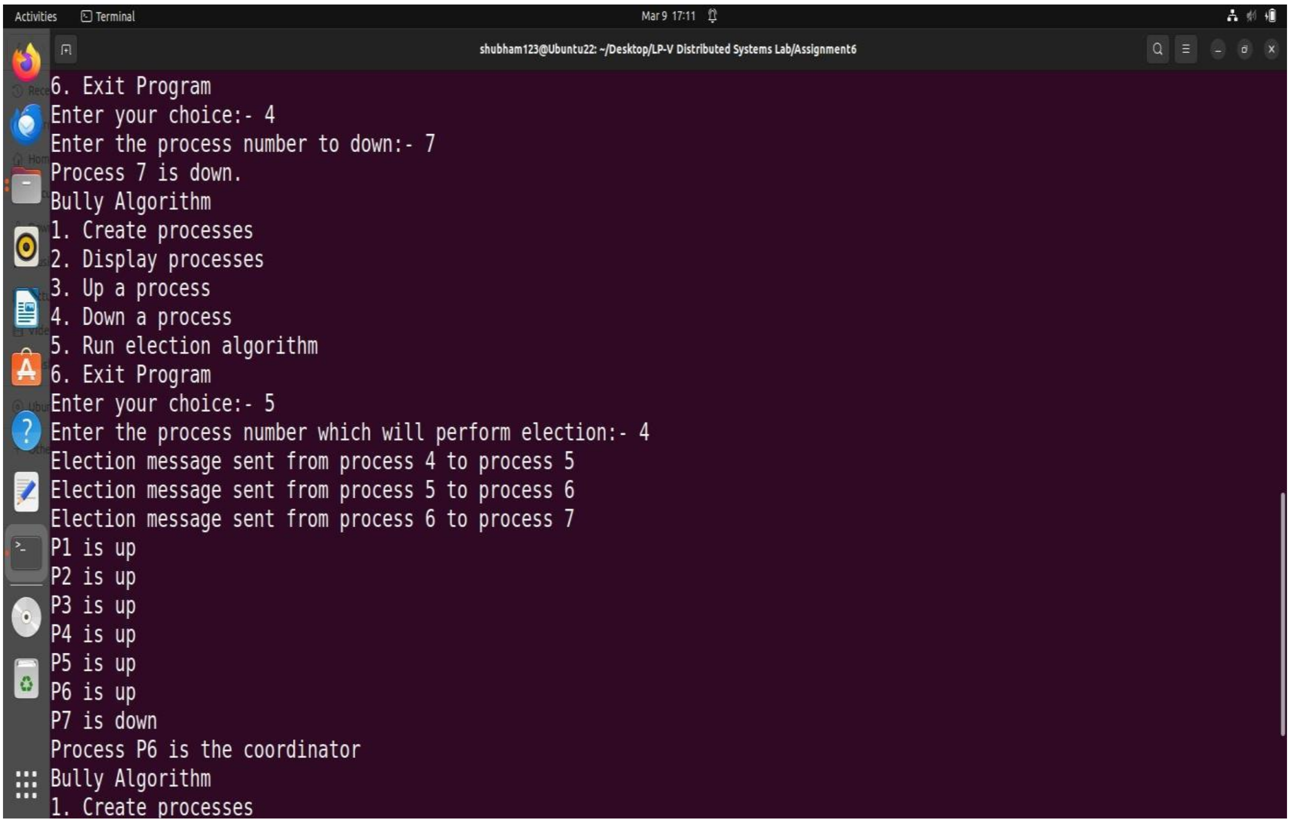
Bully Algorithm


Activities
Terminal
Mar 9 17:11
shubham123@Ubuntu22: ~/Desktop/LP-V Distributed Systems Lab/Assignment6

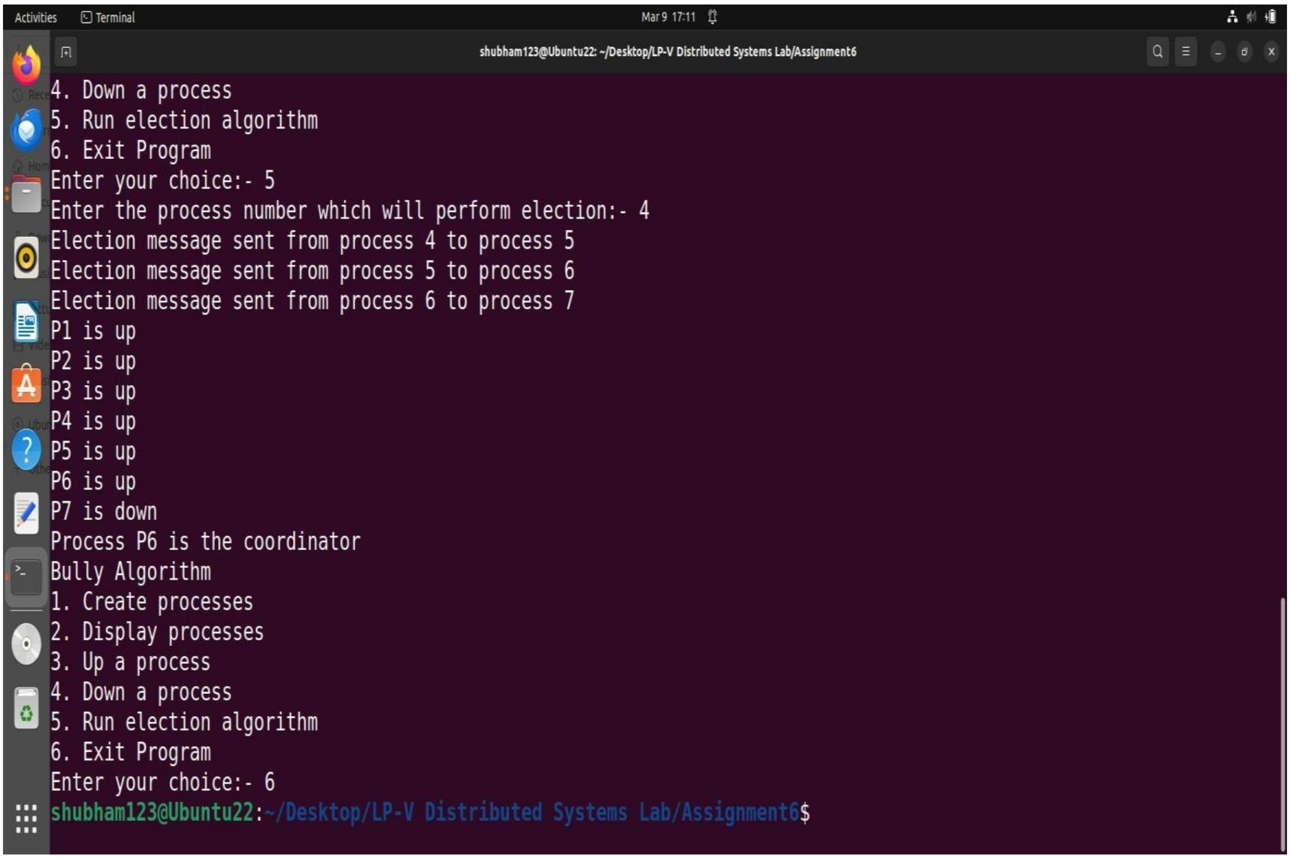
```
shubham123@Ubuntu22:~/Desktop/LP-V Distributed Systems Lab/Assignment6$ javac Bully.java
shubham123@Ubuntu22:~/Desktop/LP-V Distributed Systems Lab/Assignment6$ java Bully
Bully Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 1
Enter the number of processes:- 7
Creating processes..
P1 created
P2 created
P3 created
P4 created
P5 created
P6 created
P7 created
Process P7 is the coordinator
Bully Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
```


Activities
Terminal
Mar 9 17:11
shubham123@Ubuntu22: ~/Desktop/LP-V Distributed Systems Lab/Assignment6

```
P7 created
Process P7 is the coordinator
Bully Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 2
P1 is up
P2 is up
P3 is up
P4 is up
P5 is up
P6 is up
P7 is up
Process P7 is the coordinator
Bully Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 4
```


Activities
Terminal
Mar 9 17:11
shubham123@Ubuntu22: ~/Desktop/LP-V Distributed Systems Lab/Assignment6

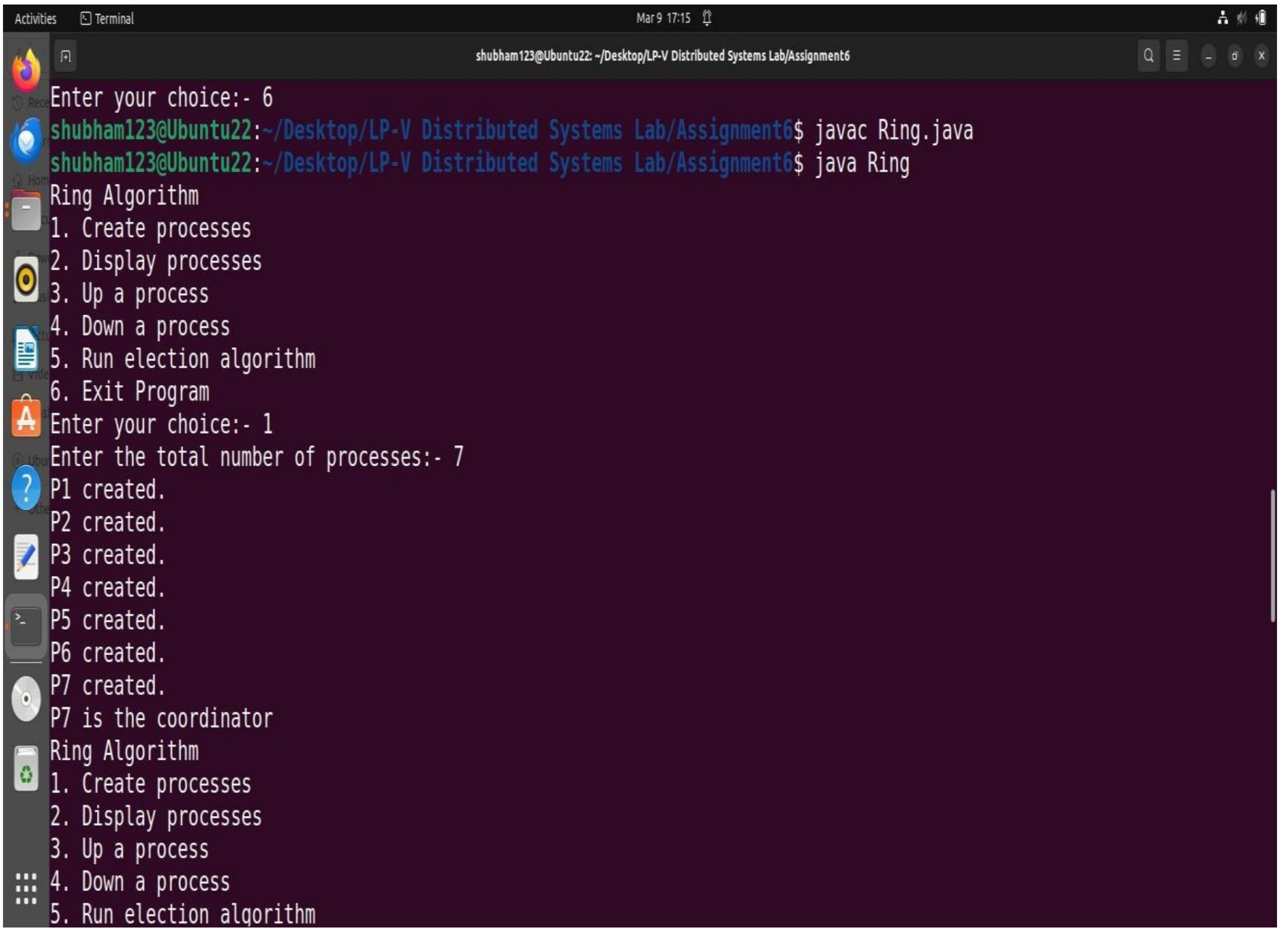
```
6. Exit Program
Enter your choice:- 4
Enter the process number to down:- 7
Process 7 is down.
Bully Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 5
Enter the process number which will perform election:- 4
Election message sent from process 4 to process 5
Election message sent from process 5 to process 6
Election message sent from process 6 to process 7
P1 is up
P2 is up
P3 is up
P4 is up
P5 is up
P6 is up
P7 is down
Process P6 is the coordinator
Bully Algorithm
1. Create processes
```


Activities
Terminal
Mar 9 17:11
shubham123@Ubuntu22: ~/Desktop/LP-V Distributed Systems Lab/Assignment6

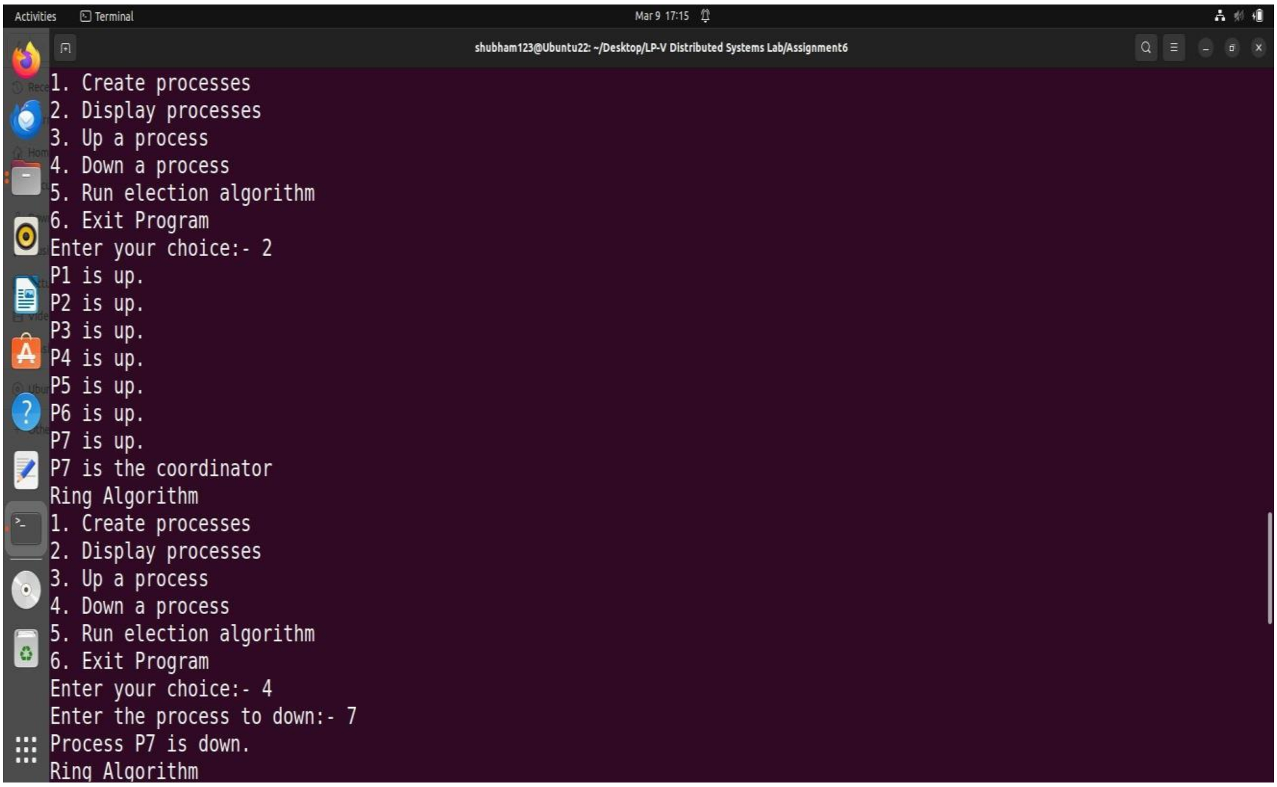
```
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 5
Enter the process number which will perform election:- 4
Election message sent from process 4 to process 5
Election message sent from process 5 to process 6
Election message sent from process 6 to process 7
P1 is up
P2 is up
P3 is up
P4 is up
P5 is up
P6 is up
P7 is down
Process P6 is the coordinator
Bully Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 6
shubham123@Ubuntu22:~/Desktop/LP-V Distributed Systems Lab/Assignment6$
```

Name: Harsh Chaudhari Class: BE-10
Batch: P-10 Roll No: 43215

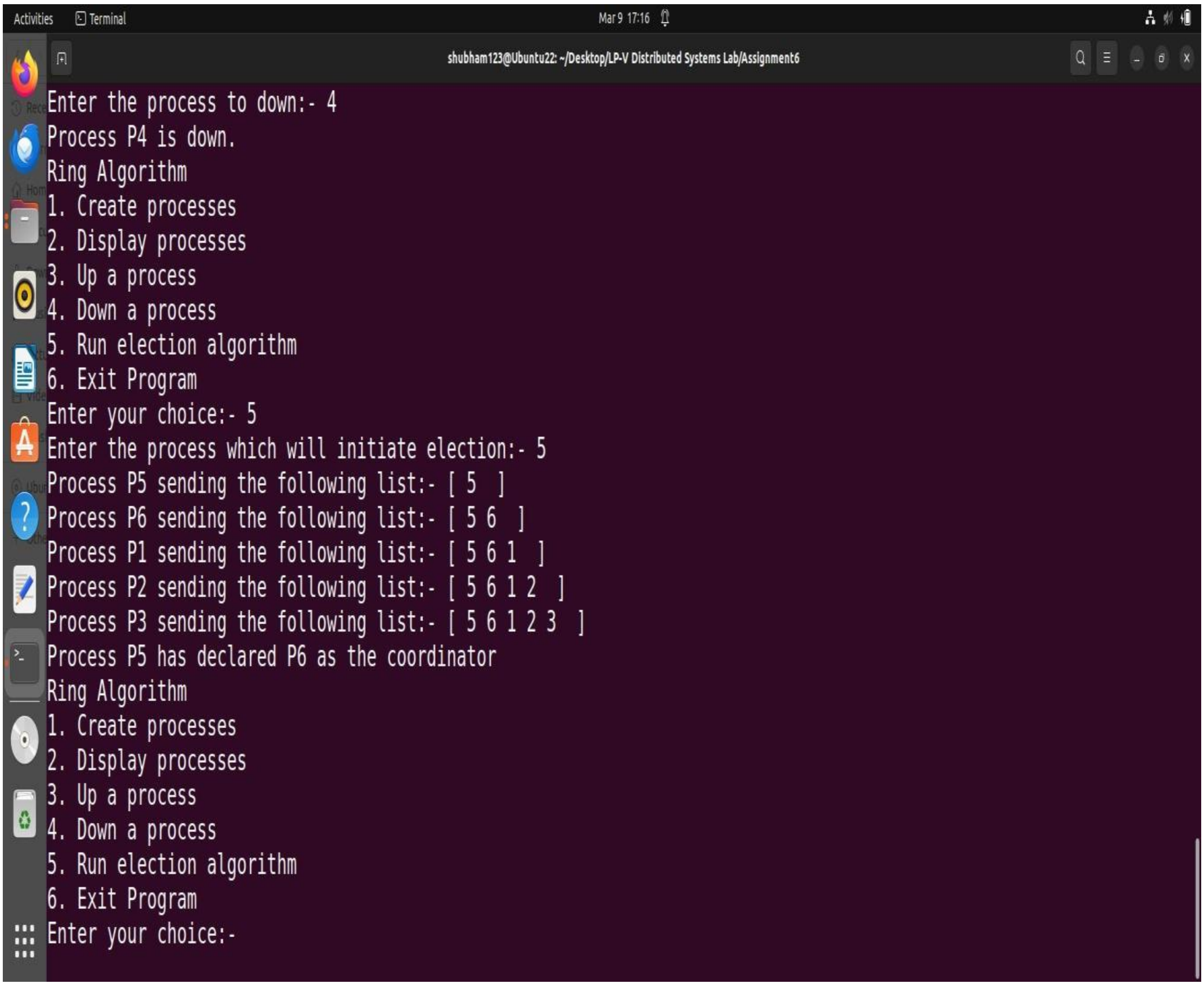
Ring Algorithm


Activities
Terminal
Mar 9 17:15
shubham123@Ubuntu22: ~/Desktop/LP-V Distributed Systems Lab/Assignment6

```
Enter your choice:- 6
shubham123@Ubuntu22:~/Desktop/LP-V Distributed Systems Lab/Assignment6$ javac Ring.java
shubham123@Ubuntu22:~/Desktop/LP-V Distributed Systems Lab/Assignment6$ java Ring
Ring Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 1
Enter the total number of processes:- 7
P1 created.
P2 created.
P3 created.
P4 created.
P5 created.
P6 created.
P7 created.
P7 is the coordinator
Ring Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
```


Activities
Terminal
Mar 9 17:15
shubham123@Ubuntu22: ~/Desktop/LP-V Distributed Systems Lab/Assignment6

```
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 2
P1 is up.
P2 is up.
P3 is up.
P4 is up.
P5 is up.
P6 is up.
P7 is up.
P7 is the coordinator
Ring Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 4
Enter the process to down:- 7
Process P7 is down.
Ring Algorithm
```


Activities
Terminal
Mar 9 17:16
shubham123@Ubuntu22: ~/Desktop/LP-V Distributed Systems Lab/Assignment6

```
Enter the process to down:- 4
Process P4 is down.
Ring Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:- 5
Enter the process which will initiate election:- 5
Process P5 sending the following list:- [ 5  ]
Process P6 sending the following list:- [ 5 6  ]
Process P1 sending the following list:- [ 5 6 1  ]
Process P2 sending the following list:- [ 5 6 1 2  ]
Process P3 sending the following list:- [ 5 6 1 2 3  ]
Process P5 has declared P6 as the coordinator
Ring Algorithm
1. Create processes
2. Display processes
3. Up a process
4. Down a process
5. Run election algorithm
6. Exit Program
Enter your choice:-
```

Conclusion:

Election algorithms are designed to choose a coordinator. We have two election algorithms for two different configurations of distributed system.

The Bully algorithm applies to system where every process can send a message to every other process in the system .

The Ring algorithm applies to systems organized as a ring (logically or physically). In this algorithm we assume that the link between the process are unidirectional and every process can message to the process on its right only.