LP-V

Pre-requisites:

1) Install JDK-8

```
sudo apt-get remove openjdk*
sudo apt update
sudo apt install openjdk-8-jdk openjdk-8-jre
```

Do this Only if any error occured

Assignment 2

Title : Remote Method Invocation

Program :

Interface - AddServerIntf.java

```
import java.rmi.*;
public interface AddServerIntf extends Remote {
public int add(int d1, int d2) throws RemoteException;
}
```

Class – AddServerImpl.java

```
import java.rmi.*;
import java.rmi.server.*;
public class AddServerImpl extends UnicastRemoteObject
implements AddServerIntf {
public AddServerImpl() throws RemoteException {
super();}
public int add(int d1, int d2) throws RemoteException {
return d1 + d2;
}
}
```

Class – AddServer.java

```
import java.net.*;
import java.rmi.*;
public class AddServer {
public static void main(String args[]) {
try {
AddServerImpl addServerImpl = new AddServerImpl();
Naming.rebind("AddServer", addServerImpl);
System.out.println("Server Started");
}
catch(Exception e) {
System.out.println("Exception: " + e);
}
}
}
```

Class- AddClient.java

```
import java.rmi.*;
import java.util.Scanner;

public class AddClient {
   public static void main(String args[]) throws Exception {
      // Create a scanner object to read input from the user
      Scanner scanner = new Scanner(System.in);
```

```
        // Prompt the user to enter two numbers
        System.out.print("Enter the first number: ");
        int num1 = scanner.nextInt();

        System.out.print("Enter the second number: ");
        int num2 = scanner.nextInt();

        // Close the scanner
        scanner.close();

        // Perform the RMI lookup and addition
        AddServerIntf obj = (AddServerIntf)Naming.lookup("AddServer");
        int result = obj.add(num1, num2);

        // Display the result
        System.out.println("Addition: " + result);
    }
}
```

Output :

1st terminal :

```
kunal@Kunal:~/DS/Assignment 1 and 2$ javac *java
kunal@Kunal:~/DS/Assignment 1 and 2$ rmiregistry
```

2nd terminal

```
kunal@Kunal:~/DS/Assignment 1 and 2$ java AddServer
Server Started
```

3rd terminal

```
kunal@Kunal:~/DS/Assignment 1 and 2$ java AddClient
Enter the first number: 25
Enter the second number: 25
Addition: 50
kunal@Kunal:~/DS/Assignment 1 and 2$
```

Commands

1St Teminal

javac *java
rmiregistry

2nd Terminal

java AddServer

3rd Terminal

java AddClient

Assignment 3

Title : Common Object Request Broker Architecture (CORBA)

Program :

ReverseClient.java

```
// client


import ReverseModule.*;

import org.omg.CosNaming.*;

import org.omg.CosNaming.NamingContextPackage.*;

import org.omg. CORBA.*;

import java.io.*;


class ReverseClient

{

	public static void main(String args[])

	{

		Reverse ReverseImpl=null;

		try

		{

		// initialize the ORB

		org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args, null);

		org.omg.CORBA.Object objRef = orb.resolve_initial_references
("NameService");

		NamingContextExt ncRef = NamingContextExtHelper.narrow (objRef);

		String name = "Reverse";

		ReverseImpl = ReverseHelper.narrow(ncRef.resolve_str(name));
```

```
                System.out.println("Enter String=");

                BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));

                String str = br.readLine();

                String tempStr = ReverseImpl.reverse_string(str);

                System.out.println(tempStr);

                }



                catch (Exception e)

                {

                e.printStackTrace();

                }

        }

}
```

ReverseImpl.java

```
import ReverseModule.ReversePOA;

import java.lang.String;

class ReverseImpl extends ReversePOA

{

        ReverseImpl()

        {

                super();

                System.out.println("Reverse Object Created");

        }

        public String reverse_string(String name)
```

```
        {
                StringBuffer str=new StringBuffer(name);
                str.reverse();
                return (("Server Send "+str));
        }
}
```

ReverseServer.java

```
import ReverseModule.*;

import org.omg.CosNaming.*;

import org.omg.CosNaming.NamingContextPackage.*;

import org.omg.CORBA.*;

import org.omg.PortableServer.*;



class ReverseServer

{

        public static void main(String[] args)

        {

                try

                {       // initialize the ORB

                        org.omg.CORBA. ORB orb = org.omg.CORBA.ORB.init(args, null);



                        // initialize the BOA/POA

                        POA rootPOA =
POAHelper.narrow(orb.resolve_initial_references("RootPOA"));
```

```
rootPOA.the_POAManager().activate();

// creating the calculator object

ReverseImpl rvr = new ReverseImpl();


// get the object reference from the servant class

org.omg.CORBA.Object ref = rootPOA.servant_to_reference(rvr);


System.out.println("Step1");

Reverse h_ref = ReverseModule.ReverseHelper.narrow(ref);

System.out.println("Step2");


org.omg.CORBA.Object objRef =
orb.resolve_initial_references("NameService");


System.out.println("Step3");

NamingContextExt ncRef =
NamingContextExtHelper.narrow(objRef);


System.out.println("Step4");


String name = "Reverse";

NameComponent path[] = ncRef.to_name(name);

ncRef.rebind(path,h_ref);
```

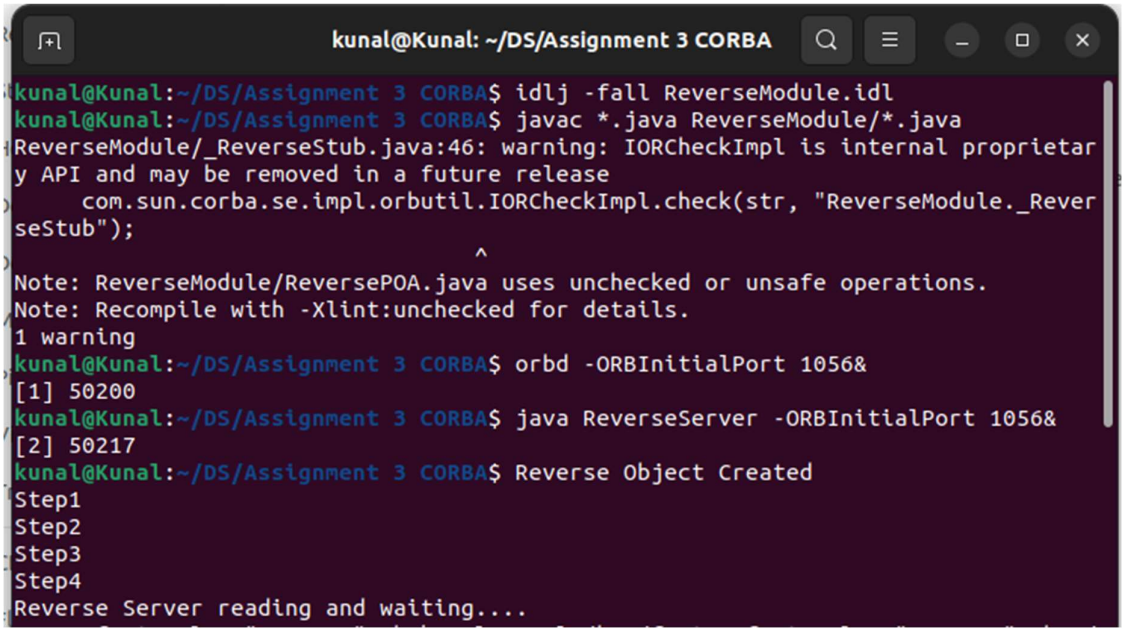
```
                System.out.println("Reverse Server reading and waiting....");

                orb.run();

            }

            catch (Exception e)

            {

                e.printStackTrace();

            }

        }

}
```

ReverseModule.idl

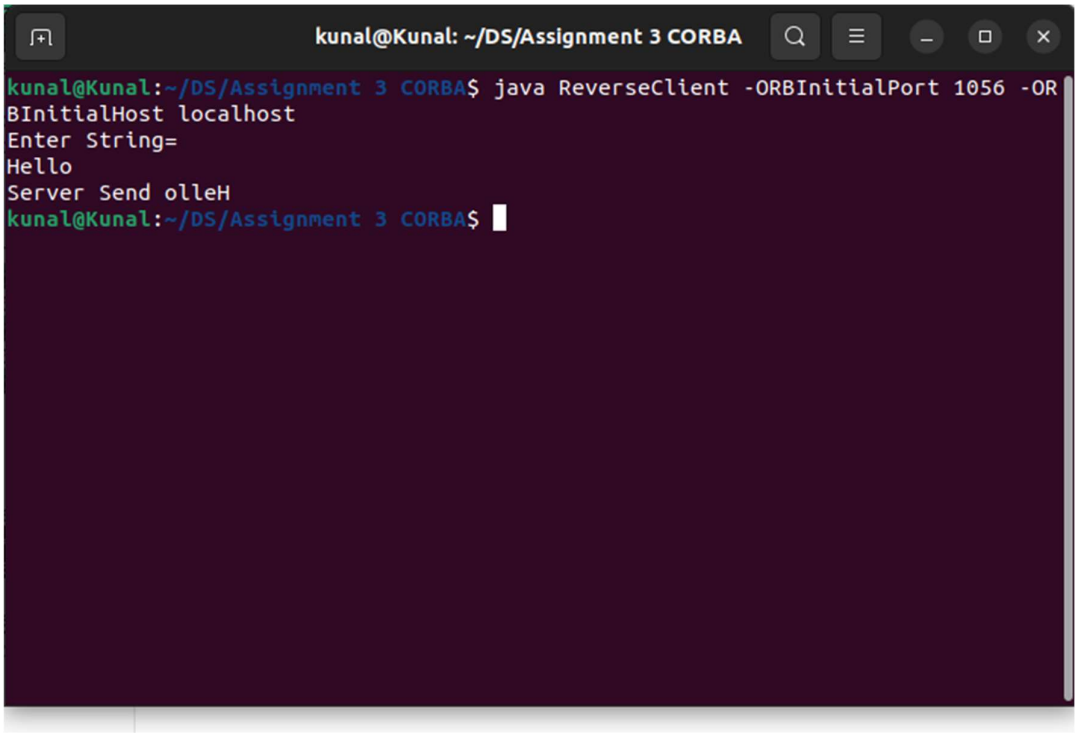
```
module ReverseModule
{
interface Reverse
{
        string reverse_string(in string str);
};
};
```

Output :

1st Terminal



2nd terminal



Commands :

1st Teminal

idlj -fall ReverseModule.idl
javac *.java ReverseModule/*.java
orbd -ORBInitialPort 1056&
java ReverseServer -ORBInitialPort 1056&

2nd Terminal

java ReverseClient -ORBInitialPort 1056 -ORBInitialHost localhost

Assignment 4

Title : Message Passing Interface (MPI)

Program :
ScatterGather.java

```
 import mpi.MPI;

   public class ScatterGather {
          public static void main(String args[]){
         //Initialize MPI execution environment
          MPI.Init(args);
         //Get the id of the process
          int rank = MPI.COMM_WORLD.Rank();
         //total number of processes is stored in size
          int size = MPI.COMM_WORLD.Size();
          int root=0;
         //array which will be filled with data by root process
          int sendbuf[]=null;

          sendbuf= new int[size];

         //creates data to be scattered
          if(rank==root){
                    sendbuf[0] = 10;
                    sendbuf[1] = 20;
                    sendbuf[2] = 30;
                    sendbuf[3] = 40;

                   //print current process number
                    System.out.print("Processor "+rank+" has data: ");
                    for(int i = 0; i < size; i++){
                              System.out.print(sendbuf[i]+" ");
                    }
                    System.out.println();
          }
         //collect data in recvbuf
          int recvbuf[] = new int[1];

         //following are the args of Scatter method
         //send, offset, chunk_count, chunk_data_type, recv, offset, chunk_count,
chunk_data_type, root_process_id
          MPI.COMM_WORLD.Scatter(sendbuf, 0, 1, MPI.INT, recvbuf, 0, 1, MPI.INT,
root);
          System.out.println("Processor "+rank+" has data: "+recvbuf[0]);
          System.out.println("Processor "+rank+" is doubling the data");
```

```
            recvbuf[0]=recvbuf[0]*2;
           //following are the args of Gather method
           //Object sendbuf, int sendoffset, int sendcount, Datatype sendtype,
//Object recvbuf, int recvoffset, int recvcount, Datatype recvtype,
//int root)
            MPI.COMM_WORLD.Gather(recvbuf, 0, 1, MPI.INT, sendbuf, 0, 1, MPI.INT, root);
           //display the gathered result
            if(rank==root){
                    System.out.println("Process 0 has data: ");
                     for(int i=0;i<4;i++){
                             System.out.print(sendbuf[i]+ " ");
                     }
            }
           //Terminate MPI execution environment
            MPI.Finalize();
    }
}
```

Output :

```
kunal@Kunal:~/DS/Assignment4MPI$ export
MPJ_HOME="/home/kunal/DS/Assignment4MPI/mpj-v0_44"
kunal@Kunal:~/DS/Assignment4MPI$ javac -cp $MPJ_HOME/lib/mpj.jar
ScatterGather.java
kunal@Kunal:~/DS/Assignment4MPI$ $MPJ_HOME/bin/mpjrun.sh -np 4 ScatterGather
MPJ Express (0.44) is started in the multicore configuration
Processor 0 has data: 10 20 30 40
Processor 0 has data: 10
Processor 0 is doubling the data
Processor 1 has data: 20
Processor 1 is doubling the data
Processor 2 has data: 30
Processor 2 is doubling the data
Processor 3 has data: 40
Processor 3 is doubling the data
Process 0 has data:
20 40 60 80
kunal@Kunal:~/DS/Assignment4MPI$
```

Commads

1) Download MPJ file and extract it (Download Link = https://sourceforge.net/projects/mpjexpress/files/releases/mpj-v0_44.tar.gz/download)

2) Command to Set a environment variable path
export MPJ_HOME=<”path to mpj-v0_44”>

to copy path right click on extracted folder and copy then paste in terminal “”

3) write a ScatterGather.java program and save it and then open the folder in terminal

4) command to compile =

```
javac -cp $MPJ_HOME/lib/mpj.jar ScatterGather.java
```

5) command to run =

```
$MPJ_HOME/bin/mpjrun.sh -np 4 ScatterGather
```

Assignment 5

Title : Clock Synchronization

Program :

Server.java

```
import java.io.*;
import java.net.*;
public class Server {
 public static void main(String[] args) {
 try {
 ServerSocket serverSocket = new ServerSocket(1234);
 System.out.println("Server started and listening on port 1234");
 while (true) {
 Socket clienSocket = serverSocket.accept();
 System.out.println("Client connected : 0 " +
clienSocket.getInetAddress().getHostAddress());
 Thread t = new Thread(new ClientHandler(clienSocket));
 t.start();
 }
 } catch (IOException e) {
 e.printStackTrace();
 }
 }
}
class ClientHandler implements Runnable {
 private Socket clientSocket;
 public ClientHandler(Socket socket) {
 this.clientSocket = socket;
 }
 @Override
 public void run() {
 try {
 PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
 BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
 out.println(System.currentTimeMillis());
 in.close();
 out.close();
 clientSocket.close();
 } catch (IOException e) {
 e.printStackTrace();
 }
 }
}
```

Client.java

```
import java.io.*;
import java.net.*;
public class Client {
 public static void main(String[] args) {
 try {
 Socket socket = new Socket("localhost", 1234);
 BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
 PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
 float serverTime = Float.parseFloat(in.readLine());
 float clientTime = System.currentTimeMillis();
 float timeDifference = serverTime - clientTime;
 System.out.println("Server TIme : " + serverTime);
 System.out.println("Client time : " + clientTime);
 System.out.println("Time Difference : " + timeDifference);
 in.close();
 out.close();
 socket.close();
 } catch (IOException e) {
 e.printStackTrace();
 }
 }
}
```

Output :

1st Terminal

```
kunal@Kunal:~/DS/Assignment 5 Clock Synchronization$ javac *.java
kunal@Kunal:~/DS/Assignment 5 Clock Synchronization$ java Server
Server started and listening on port 1234
Client connected : 0 127.0.0.1
```

2nd terminal

```
kunal@Kunal:~/DS/Assignment 5 Clock Synchronization$ java Client
Server TIme : 1.71372262E12
Client time : 1.71372262E12
Time Difference : 0.0
kunal@Kunal:~/DS/Assignment 5 Clock Synchronization$
```

Commands

1st Terminal

```
javac *.java
java Server
```

2nd Terminal

```
java Client
```

Assignment 6

Title : Mutual Exclusion

Program :

Tokering.java

```
import java.util.*;
class tokenring {
 public static void main(String args[]) throws Throwable {
 Scanner scan = new Scanner(System.in);
 System.out.println("Enter no of Nodes:");
 int n = scan.nextInt();
 int m = n - 1;
 int token = 0;
 int ch = 0, flag = 0;
 for (int i = 0; i < n; i++) {
 System.out.print("" + i);
 }
 System.out.println("" + 0);
 do {
 System.out.println("Enter sender:");
 int s = scan.nextInt();
 System.out.println("Enter receiver:");
 int r = scan.nextInt();
 System.out.println("Enter Data:");
 int a;
 a = scan.nextInt();
 System.out.print("Token Passing");
 for (int i = token, j = token; (i % n) != s; i++, j = (j + 1) % n) {
 System.out.print("" + j + "->");
 }
 System.out.println("" + s);
 System.out.println("Sender" + s + "Sending Data:" + a);
 for (int i = s + 1; i != r; i = (i + 1) % n) {
 System.out.println("Data" + a + "Forwarded By:" + i);
 }
 System.out.println("Receiver" + r + "Received Data:" + a + "\n");
 token = s;
 do {
 try {
 if (flag == 1)
 System.out.print("Invalid Input!!...");
 System.out.print("Do you want to send again?? Enter 1 for yes and 0 for No:");
 ch = scan.nextInt();
 if (ch != 1 && ch != 0)
 flag = 1;
 else
```

```
 flag = 0;
 } catch (InputMismatchException e) {
 System.out.println("Invalid Input");
 }
 } while (ch != 1 && ch != 0);
 } while (ch == 1);
 }
}
```

Output :

```
kunal@Kunal:~/DS/Assignment 6 Mutual Exclusion$ javac tokenring.java
kunal@Kunal:~/DS/Assignment 6 Mutual Exclusion$ java tokenring
Enter no of Nodes:
6
0123450
Enter sender:
2
Enter receiver:
5
Enter Data:
1
Token Passing0->1->2
Sender2Sending Data:1
Data1Forwarded By:3
Data1Forwarded By:4
Receiver5Received Data:1

Do you want to send again?? Enter 1 for yes and 0 for No:0
kunal@Kunal:~/DS/Assignment 6 Mutual Exclusion$
```

Commands

1st terminal
javac tokenring.java
java tokenring

Assignment 7

Title : Election Algorithms

Program :
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;
    }
  }
 }
}
```

Output :

```
kunal@Kunal:~/DS/Assignment 7 Election Algorithm$ javac Bully.java
kunal@Kunal:~/DS/Assignment 7 Election Algorithm$ 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:- 5
Creating processes..
```

```
P1 created
P2 created
P3 created
P4 created
P5 created
Process P5 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
Enter the process number to down:- 5
Process 5 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:- 2
P1 is up
P2 is up
P3 is up
P4 is up
P5 is down
Process P5 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:- 5
Enter the process number which will perform election:- 4
Election message sent from process 4 to process 5
P1 is up
P2 is up
P3 is up
P4 is up
P5 is down
Process P4 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
kunal@Kunal:~/DS/Assignment 7 Election Algorithm$
```

Commands

```
javac Bully.java
java Bully
```

Program :

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 :

```
kunal@Kunal:~/DS/Assignment 7 Election Algorithm$ javac Ring.java
kunal@Kunal:~/DS/Assignment 7 Election Algorithm$ 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:- 5
P1 created.
P2 created.
P3 created.
P4 created.
P5 created.
P5 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:- 5
Process P5 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:- 2
P1 is up.
P2 is up.
P3 is up.
P4 is up.
P5 is down.
P5 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:- 5
Enter the process which will initiate election:- 2
Process P2 sending the following list:- [ 2  ]
Process P3 sending the following list:- [ 2 3  ]
Process P4 sending the following list:- [ 2 3 4  ]
Process P1 sending the following list:- [ 2 3 4 1  ]
Process P2 has declared P4 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:- 6
kunal@Kunal:~/DS/Assignment 7 Election Algorithm$
```

Commands

```
javac Ring.java
java Ring
```