Student Name :- Dhavale Mayur Vitthal
Roll No: 505
Assignment No:- 06
TITLE : BULLY AND RING ALGORITHM FOR LEADER ELECTION
==========

RING ALGORITHM CODE:

```
import java.util.ArrayList;
import java.util.Scanner;

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

    public Ring(int max) {
        coordinator = max;
        max_processes = max;
        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++) {
            System.out.println("P" + (i + 1) + " is " + (processes[i] ? "up." :
"down."));
        }
        System.out.println("P" + coordinator + " is the coordinator");
    }

    void initElection(int process_id) {
        ArrayList<Integer> pid = new ArrayList<>();
        int temp = process_id;

        do {
            if (processes[temp - 1]) {
                pid.add(temp);
                System.out.print("Process P" + temp + " sending the following
list:- " + pid + "\n");
            }
            temp = (temp % max_processes) + 1;
        } while (temp != process_id);

        coordinator = pid.stream().max(Integer::compare).orElse(coordinator);
```

```
        System.out.println("Process P" + process_id + " has declared P" +
coordinator + " as the coordinator");
    }

    public static void main(String args[]) {
        Ring ring = null;
        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. Run election algorithm");
            System.out.println("4. Exit Program");
            System.out.print("Enter your choice:- ");
            choice = sc.nextInt();

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

            // Ask user which process crashed or is not responding
            System.out.print("Enter the process number that has crashed or is not
responding: ");
            int crashed_process = sc.nextInt();
            ring.processes[crashed_process - 1] = false;
            System.out.println("Process P" + crashed_process + " has crashed or is
not responding.");

            // Ask for the initiator of the election
```

```
            System.out.print("Enter the process which will initiate election: ");
            int election_initiator = sc.nextInt();
            ring.initElection(election_initiator);
        }
    }
}
```

==========

```
//OUTPUT

Ring Algorithm
1. Create processes
2. Display processes
3. Run election algorithm
4. 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
Enter the process number that has crashed or is not responding: 5
Process P5 has crashed or is not responding.
Enter the process which will initiate election: 4
Process P4 sending the following list:- [4]
Process P6 sending the following list:- [4, 6]
Process P7 sending the following list:- [4, 6, 7]
Process P1 sending the following list:- [4, 6, 7, 1]
Process P2 sending the following list:- [4, 6, 7, 1, 2]
Process P3 sending the following list:- [4, 6, 7, 1, 2, 3]
Process P4 has declared P7 as the coordinator
Ring Algorithm
1. Create processes
2. Display processes
3. Run election algorithm
4. Exit Program
Enter your choice:- 4
```

Bully Algorithm -Code:

```
// import required classes and packages
import java.util.Scanner;

// create process class for creating a process having id and status
class Process {
```

```
    // declare variables
    public int id;
    public String status;

    // initialize variables using constructor
    public Process(int id) {
        this.id = id;
        this.status = "active";
    }
}

// create class BullyAlgoExample2 for understanding the concept of Bully algorithm
public class BullyAlgoExample2 {

    // initialize variables and array
    Scanner sc;
    Process[] processes;
    int n;

    // initialize Scanner class object in constructor
    public BullyAlgoExample2() {
        sc = new Scanner(System.in);
    }

    // create ring() method for initializing the ring
    public void ring() {

        // get input from the user for processes
        System.out.println("Enter total number of processes");
        n = sc.nextInt();

        // initialize processes array
        processes = new Process[n];
        for (int i = 0; i < n; i++) {
            processes[i] = new Process(i);
        }
    }

    // method to take user input for failed process ID
    public int inputFailedProcess() {
        System.out.println("Enter the ID of the process that has failed:");
        return sc.nextInt();
    }

    // create election() method for electing process
    public void performElection(int failedProcessId) {

        // show failed process
        System.out.println("Process having id " + failedProcessId + " fails");
```

```
        // change status to Inactive of the failed process
        processes[failedProcessId].status = "Inactive";

        // declare and initialize variables
        int idOfInitiator = 0;
        boolean overStatus = true;

        // use while loop to repeat steps
        while (overStatus) {

            boolean higherProcesses = false;

            // iterate all the processes
            for (int i = idOfInitiator + 1; i < n; i++) {
                if (processes[i].status.equals("active")) {
                    System.out.println("Process " + idOfInitiator + " Passes
Election(" + idOfInitiator + ") message to process" + i);
                    higherProcesses = true;
                }
            }

            // check for higher process
            if (higherProcesses) {

                // use for loop to again iterate processes
                for (int i = idOfInitiator + 1; i < n; i++) {
                    if (processes[i].status.equals("active")) {
                        System.out.println("Process " + i + " Passes Ok(" + i + ")
message to process" + idOfInitiator);
                    }
                }
                // increment initiator id
                idOfInitiator++;
            } else {
                // get the last process from the processes that will become
coordinator
                int coord = getMaxValue();

                // show process that becomes the coordinator
                System.out.println("Finally Process " + coord + " Becomes
Coordinator");

                for (int i = coord - 1; i >= 0; i--) {
                    if (processes[i].status.equals("active")) {
                        System.out.println("Process " + coord + " Passes
Coordinator(" + coord + ") message to process " + i);
                    }
                }

                System.out.println("End of Election");
```

```
                overStatus = false;
                break;
            }
        }

    }

    // create getMaxValue() method that returns index of max process
    public int getMaxValue() {
        int mxId = -99;
        int mxIdIndex = 0;
        for (int i = 0; i < processes.length; i++) {
            if (processes[i].status.equals("active") && processes[i].id > mxId) {
                mxId = processes[i].id;
                mxIdIndex = i;
            }
        }
        return mxIdIndex;
    }

    // main() method start
    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
        String answer;

        do {
            // create instance of the BullyAlgoExample2 class
            BullyAlgoExample2 bully = new BullyAlgoExample2();

            // call ring() method
            bully.ring();

            // take user input for failed process
            int failedProcessId = bully.inputFailedProcess();

            // perform election with the failed process ID
            bully.performElection(failedProcessId);

            System.out.println("Do you want to continue? (yes/no)");
            answer = scanner.nextLine();
        } while (answer.equalsIgnoreCase("yes"));

        scanner.close();
    }

}
```

==========

```
// OUTPUT
/*
PS C:\Users\Mayur\Desktop\New folder (3)\New folder> javac BullyAlgoExample2.java
PS C:\Users\Mayur\Desktop\New folder (3)\New folder> java BullyAlgoExample2
Enter total number of processes
5
Enter the ID of the process that has failed:
3
Process having id 3 fails
Process 0 Passes Election(0) message to process1
Process 0 Passes Election(0) message to process2
Process 0 Passes Election(0) message to process4
Process 1 Passes Ok(1) message to process0
Process 2 Passes Ok(2) message to process0
Process 4 Passes Ok(4) message to process0
Process 1 Passes Election(1) message to process2
Process 1 Passes Election(1) message to process4
Process 2 Passes Ok(2) message to process1
Process 4 Passes Ok(4) message to process1
Process 2 Passes Election(2) message to process4
Process 4 Passes Ok(4) message to process2
Process 3 Passes Election(3) message to process4
Process 4 Passes Ok(4) message to process3
Finally Process 4 Becomes Coordinator
Process 4 Passes Coordinator(4) message to process 2
Process 4 Passes Coordinator(4) message to process 1
Process 4 Passes Coordinator(4) message to process 0
End of Election
Do you want to continue? (yes/no)
yes
Enter total number of processes
7
Enter the ID of the process that has failed:
5
Process having id 5 fails
Process 0 Passes Election(0) message to process1
Process 0 Passes Election(0) message to process2
Process 0 Passes Election(0) message to process3
Process 0 Passes Election(0) message to process4
Process 0 Passes Election(0) message to process6
Process 1 Passes Ok(1) message to process0
Process 2 Passes Ok(2) message to process0
Process 3 Passes Ok(3) message to process0
Process 4 Passes Ok(4) message to process0
Process 6 Passes Ok(6) message to process0
Process 1 Passes Election(1) message to process2
Process 1 Passes Election(1) message to process3
Process 1 Passes Election(1) message to process4
Process 1 Passes Election(1) message to process6
Process 2 Passes Ok(2) message to process1
```

```
Process 3 Passes Ok(3) message to process1
Process 4 Passes Ok(4) message to process1
Process 6 Passes Ok(6) message to process1
Process 2 Passes Election(2) message to process3
Process 2 Passes Election(2) message to process4
Process 2 Passes Election(2) message to process6
Process 3 Passes Ok(3) message to process2
Process 4 Passes Ok(4) message to process2
Process 6 Passes Ok(6) message to process2
Process 3 Passes Election(3) message to process4
Process 3 Passes Election(3) message to process6
Process 4 Passes Ok(4) message to process3
Process 6 Passes Ok(6) message to process3
Process 4 Passes Election(4) message to process6
Process 6 Passes Ok(6) message to process4
Process 5 Passes Election(5) message to process6
Process 6 Passes Ok(6) message to process5
Finally Process 6 Becomes Coordinator
Process 6 Passes Coordinator(6) message to process 4
Process 6 Passes Coordinator(6) message to process 3
Process 6 Passes Coordinator(6) message to process 2
Process 6 Passes Coordinator(6) message to process 1
Process 6 Passes Coordinator(6) message to process 0
End of Election
Do you want to continue? (yes/no)
no
PS C:\Users\Mayur\Desktop\New folder (3)\New folder>


 */
```