Name: Sayyed Sohail Rashid	Course Name: DC-LAB
Class: BE-CO	Batch: 01
Roll no: 18CO48	Experiment No: 05

Aim **:** To Implement the Bully Algorithm.

Code**:**

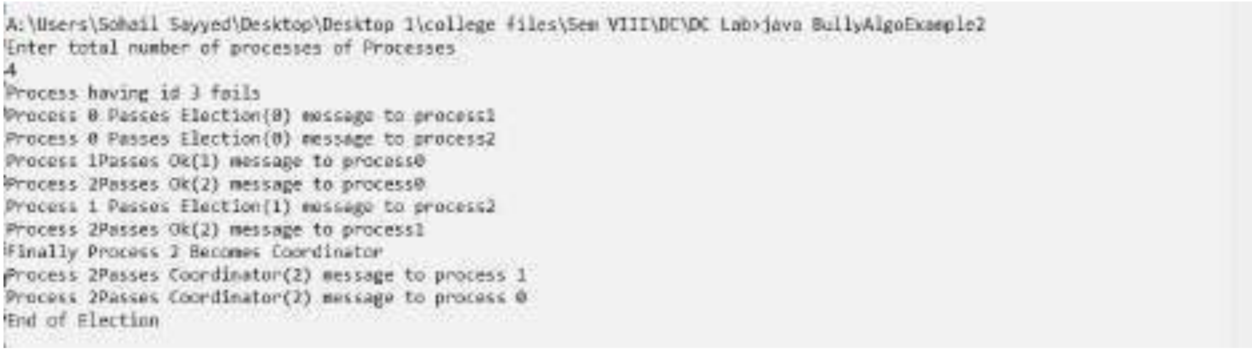
BullyAlgoExample2.java

```
    import java.util.Scanner;

// create process class for creating a process having id and statusclass
Process{
    // declare variables
    public int id;
    public String status;

    // initialize variables using constructorpublic
    Process(int id){
        this.id = id; this.status
        = "active";
    }
}
// create class BullyAlgoExample2 for understanding the concept of Bully algorithmpublic class
BullyAlgoExample2 {

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

    // initialize Scanner class object in constructorpublic
    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 of Processes");n =
            sc.nextInt();

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

        // create election() method for electing processpublic
        void performElection(){

            // we use the sleep() method to stop the execution of the current threadtry {
                Thread.sleep(1000);
            } catch (InterruptedException e) {

                e.printStackTrace();
            }

            // show failed process
            System.out.println("Process having id "+processes[getMaxValue()].id+" fails");

            // change status to Inactive of the failed process
            processes[getMaxValue()].status = "Inactive";

            // declare and initialize variablesint
            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 == "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 == "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 coordinatorint coord
                    = processes[getMaxValue()].id;

                    // 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 == "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 processpublic int
    getMaxValue(){
        int mxId = -99;
        int mxIdIndex = 0;
        for(int i = 0; i<processes.length; i++){
            if(processes[i].status == "active" && processes[i].id >mxId){mxId =
                processes[i].id;
                mxIdIndex = i;
            }
        }
        return mxIdIndex;
    }

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

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

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

    }

}
```



```
A:\Users\Sohail Sayyed\Desktop\Desktop 1\college files\Sem VIII\DC\DC Lab>java BullyAlgoExample2
Enter total number of processes of Processes
4
Process having id 3 fails
Process 0 Passes Election(0) message to process1
Process 0 Passes Election(0) message to process2
Process 1Passes Ok(1) message to process0
Process 2Passes Ok(2) message to process0
Process 1 Passes Election(1) message to process2
Process 2Passes Ok(2) message to process1
Finally Process 2 Becomes Coordinator
Process 2Passes Coordinator(2) message to process 1
Process 2Passes Coordinator(2) message to process 0
End of Election
```

Conclusion:

Bully Algorithm has been executed successfully and gives the required output.