



Code

1. Bully.java

J Bully.java >  Bully >  up(int)

```
1
2 import java.util.Scanner;
3 public class Bully {
4
5     static boolean state[] = new boolean[5];
6     int coordinator;
7     public static void up(int up)
8     {
9         if(state[up-1]==true)
10        {
11            System.out.println("process "+up+" is already up");
12        }
13        else
14        {
15            state[up-1] = true;
16            System.out.println("process "+up+" held election");
17            for(int i=up;i<5;i++)
18            {
19                System.out.println("election message sent from process "+up+" to process" +(i+1));
20            }
21            for(int i=up+1;i<=5;i++)
22            {
23                if(state[i-1]==true)
24                {
25                    System.out.println("alive message send from process "+i+" to process "+up);
26                    break;
27                }
28            }
29        }
30    }
31    public static void down(int down)
32    {
33        if(state[down-1]==false)
34        {
35            System.out.println("process "+down+" is already down.");
36        }
37        else
38        {
39            state[down-1] = false;
40        }
41    }
42 }
43
44 public static void mess(int mess)
45 {
46     if(state[mess-1]==true)
47     {
48         if(state[4]==true)
```

```
J Ring.java J Bully.java x
J Bully.java > Bully > up(int)
48 if(state[4]==true)
49 {
50     System.out.println("OK");
51 }
52 else
53 {
54     if(state[4]==false)
55     {
56         System.out.println("process"+mess+"election");
57         for(int i=mess;i<5;i++)
58         {
59             System.out.println("election send from process"+mess+"to process "+(i+1));
60         }
61
62         for(int i=5;i>=mess;i--)
63         {
64             if(state[i-1]==true)
65             {
66                 System.out.println("Coordinator message send from process"+i+"to all");
67                 break;
68             }
69         }
70     }
71 }
72 }
73 else
74 {
75     System.out.println("Prccess"+mess+"is down");
76 }
77 }
78
Run | Debug
79 public static void main(String[] args) {
80     Scanner sc = new Scanner(System.in);
81     int choice;
82     for(int i=0;i<5;i++)
83     {
84         state[i] = true;
85     }
86     System.out.println("5 active process are:");
87     System.out.println("Process up = p1 p2 p3 p4 p5");
88     System.out.println("Process 5 is coordinator");
89
90     do
91     {
92         System.out.println(".....");
93         System.out.println("1 up a process.");
94         System.out.println("2.down a process");
95         System.out.println("3. send a message");
```

J Ring.java

J Bully.java ×

J Bully.java > Bully > up(int)

```
92 System.out.println(".....");
93 System.out.println("1 up a process.");
94 System.out.println("2.down a process");
95 System.out.println("3 send a message");
96 System.out.println("4.Exit");
97 choice = sc.nextInt();
98 switch(choice)
99 {
100     case 1:
101     {
102         System.out.println("bring proces up");
103         int up = sc.nextInt();
104         if(up==5)
105         {
106             System.out.println("process 5 is co-ordinator");
107             state[4] = true;
108         }
109         else
110         {
111             up(up);
112         }
113     }
114     break;
115     case 2:
116     {
117         System.out.println("bring down any process.");
118         int down = sc.nextInt();
119         down(down);
120     }
121     break;
122     case 3:
123     {
124         System.out.println("which process will send message");
125         int mess = sc.nextInt();
126         mess(mess);
127     }
128     break;
129 }
130 }
131 }
132 }
133 }
134 while(choice!=4);
135 sc.close();
136 }
137 }
```

2. Ring.java

J Ring.java × J Bully.java 1

J Ring.java >  Ring >  election(int)

```
1  import java.util.Scanner;
2
3  public class Ring {
4
5      int n, inactive_count;
6      int coordinator;
7      boolean[] process_state;
8
9      public Ring(int n) {
10         this.n = n;
11         this.inactive_count = 0;
12         this.process_state = new boolean[n];
13         // State all processes as active
14         for(int i = 0; i < n; i++) {
15             this.process_state[i] = true;
16         }
17         this.coordinator = n - 1;
18         System.out.println("Process " + n + " is set as initial coordinator");
19     }
20
21     public void deactivate_process(int id) {
22         /*
23          * Input   : Process ID
24          * Utility : Deactivate process
25          * Output  : None
26          */
27         if(id > n || id < 0) {
28             System.out.println("Invalid ID");
29             return;
30         }
31         if(!process_state[id - 1]) {
32             System.out.println("Process already inactive");
33         } else {
34             process_state[id - 1] = false;
35             System.out.println("Process " + id + " deactivated");
36             inactive_count += 1;
37         }
38     }
39
40     public void view_ring() {
41         /*
42          * Input   : None
43          * Utility : Display ring
44          * Output  : Console output
45          */
46
47         if(this.inactive_count == n) {
48             System.out.println("All members inactive...");
```

```

49         return;
50     }
51     System.out.println("Active Ping members");
52     for(int i = 0; i < process_state.length; i++) {
53         if(process_state[i]) System.out.println((i + 1) + " ");
54     }
55 }
56
57 public void election(int id) {
58     /*
59      * Input    : Initiator
60      * Utility   : Hold election process to select coordinator
61      * Output    : Coordinator id
62      */
63     if(this.inactive_count == this.n) {
64         System.out.println("All members inactive...");
65         System.out.println("Aborting election process...");
66         this.coordinator = -1;
67         return;
68     }
69     id = id - 1;
70     int current_coordinator = id;
71     int token = (id + 1) % n;
72     System.out.println("\nElection initiator : " + (id + 1));
73     // Election algorithm
74     while(token != id) {
75         System.out.println("Token at process " + (token + 1));
76         if(this.process_state[token]) {
77             if(token > current_coordinator) {
78                 current_coordinator = token;
79             }
80         }
81         token = (token + 1) % this.n;
82     }
83     System.out.println("Elected coordinator : " + (current_coordinator + 1));
84     this.coordinator = current_coordinator;
85 }
86
87 public void ping_coordinator(int id) {
88     if(!this.process_state[id - 1]) {
89         System.out.println("Process inactive...");
90         System.out.println("Aborting...");
91         return;
92     }
93     if(id == coordinator) {
94         if(this.process_state[id - 1]) {
95             System.out.println("Coordinator active");
96         }
97     }

```

```

J Ring.java x J Bully.java 1
J Ring.java > Ring > election(int)
96         } else {
97             System.out.println("Coordinator inactive!\nInitiate election from other process");
98         }
99     }
100     System.out.println("Sending message from process " + id + " to " + (this.coordinator + 1));
101     if(!this.process_state[this.coordinator]) {
102         System.out.println("Coordinator process not responding");
103         System.out.println("Conducting election...");
104         this.election(id);
105     } else {
106         System.out.println("Coordinator alive");
107     }
108 }
109
110 public void setCoordinator(int c) {
111     this.coordinator = c;
112 }
113
Run | Debug
114 public static void main(String[] args) {
115     int choice = 0;
116     Scanner sc = new Scanner(System.in);
117     System.out.println("Enter number of processes: ");
118     int n = sc.nextInt();
119     Ring ring = new Ring(n);
120
121     while(choice < 5) {
122         System.out.println("*****Menu*****");
123         System.out.println("1. Deactivate a process");
124         System.out.println("2. Ping coordinator");
125         System.out.println("3. View Ring");
126         System.out.println("4. Election");
127         System.out.println("5. Exit");
128         System.out.println("*****");
129         System.out.println("Enter Choice : ");
130         choice = sc.nextInt();
131         switch(choice) {
132             case 1 : {
133                 int id;
134                 System.out.println("Enter process ID : ");
135                 id = sc.nextInt();
136                 ring.deactivate_process(id);
137                 System.out.println("");
138                 break;
139             }
140             case 2 : {
141                 int id;
142                 System.out.println("Enter process ID for sender");

```

```

142         System.out.println("Enter process ID for sender");
143         id = sc.nextInt();
144         ring.ping_coordinator(id);
145         System.out.println("");
146         break;
147     }
148     case 3 : {
149         ring.view_ring();
150         System.out.println("");
151         break;
152     }
153     case 4 : {
154         int id;
155         System.out.println("Enter process ID for election initiator");
156         id = sc.nextInt();
157         ring.election(id);
158         System.out.println("");
159         break;
160     }
161     case 5 :
162     default : {
163         System.out.println("");
164         break;
165     }
166 }
167
168 }
169 System.out.println("Program terminated..");
170 sc.close();
171 }
172 }
173

```

Output

1. Bully Algorithm

```
pict@pict-OptiPlex-5060: ~/Desktop
pict@pict-OptiPlex-5060:~/Desktop$ java Bully
5 active process are:
Process up   = p1 p2 p3 p4 p5
Process 5 is coordinator
.....
1 up a process.
2.down a process
3 send a message
4.Exit
1
bring proces up
2
process2is already up
.....
1 up a process.
2.down a process
3 send a message
4.Exit
2
bring down any process.
5
.....
1 up a process.
2.down a process
3 send a message
4.Exit
3
which process will send message
3
process3election
election send from process3to process 4
election send from process3to process 5
Coordinator message send from process4to all
.....
1 up a process.
2.down a process
3 send a message
4.Exit
1
bring proces up
4
process4is already up
.....
1 up a process.
2.down a process
3 send a message
4.Exit
4
pict@pict-OptiPlex-5060:~/Desktop$
```


2. Ring Algorithm

```
varadmash@varadmash-G3-3590:~/LP5_lab/Assignment6$ javac Ring.java
varadmash@varadmash-G3-3590:~/LP5_lab/Assignment6$ java Ring
Enter number of processes:
5
Process 5 is set as initial coordinator
*****Menu*****
1. Deactivate a process
2. Ping coordinator
3. View Ring
4. Election
5. Exit
*****
Enter Choice :
1
Enter process ID :
5
Process 5 deactivated

*****Menu*****
1. Deactivate a process
2. Ping coordinator
3. View Ring
4. Election
5. Exit
*****
Enter Choice :
2
Enter process ID for sender
3
Sending message from process 3 to 5
Coordinator process not responding
Conducting election...
```

```

Sending message from process 3 to 5
Coordinator process not responding
Conducting election...

Election initiator : 3
Token at process 4
Token at process 5
Token at process 1
Token at process 2
Elected coordinator : 4

*****Menu*****
1. Deactivate a process
2. Ping coordinator
3. View Ring
4. Election
5. Exit
*****
Enter Choice :
3
Active Ring members
1
2
3
4

*****Menu*****
1. Deactivate a process
2. Ping coordinator
3. View Ring
4. Election
5. Exit
*****
Enter Choice :
5

Program terminated..
varadmash@varadmash-G3-3590:~/LP5_lab/Assignment6$
```