

The screenshot shows an IDE with the file `Bully.java` open. The code implements a Bully algorithm with methods `up`, `down`, and `mess`. The console output shows the execution of the program, including the initial state of processes and the messages sent during the election process.

```

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

```

Console Output:

```

Bully(1) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (28-Dec-2018, 7:3
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
.....
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
2
process2election
election send from process2to process 3
election send from process2to process 4
election send from process2to process 5
Coordinator message send from process4to all
.....
1 up a process.
2 down a process
3 send a message
4.Exit

```

The screenshot shows an IDE with the file `Ring.java` open. The code implements a Ring algorithm for process coordination. The console output shows the execution of the program, including the input of process numbers and the messages sent during the election process.

```

1 import java.util.Scanner;
2
3 public class Ring {
4
5     public static void main(String[] args) {
6
7         // TODO Auto-generated method stub
8
9         int temp, i, j;
10         char str[] = new char[10];
11         Rr proc[] = new Rr[10];
12
13         // object initialisation
14         for (i = 0; i < proc.length; i++)
15             proc[i] = new Rr(i);
16
17         // scanner used for getting input from console
18         Scanner in = new Scanner(System.in);
19         System.out.println("Enter the number of process : ");
20         int num = in.nextInt();
21
22         // getting input from users
23         for (i = 0; i < num; i++) {
24             proc[i].index = i;
25             System.out.println("Enter the id of process : ");
26             proc[i].id = in.nextInt();
27             proc[i].state = "active";
28             proc[i].f = 0;
29         }
30
31         // sorting the processes from on the basis of id
32         for (i = 0; i < num - 1; i++) {
33             for (j = 0; j < num - 1; j++) {
34                 if (proc[j].id > proc[j + 1].id) {
35                     temp = proc[j].id;
36                     proc[j].id = proc[j + 1].id;
37                     proc[j + 1].id = temp;
38                 }
39             }
40         }
41     }
42 }

```

Console Output:

```

<terminated> Ring (1) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/b
Enter the number of process :
3
Enter the id of process :
5 6 8
Enter the id of process :
Enter the id of process :
[0] 5 [1] 6 [2] 8
process 8select as co-ordinator
1.election 2.quit
1
Enter the Process number who initialisied election :
2
Process 8 send message to 5
Process 5 send message to 6
Process 6 send message to 8
process 8select as co-ordinator
1.election 2.quit
2
Program terminated ...

```