ASSIGNMENT NO. 4	
Assignment No. 4	To develop any distributed algorithm for **leader election**.
Objective(s):	By the end of this assignment, the student will be able to explain the concept of Leader Election Algorithms.
Tools	Eclipse, Java 8

Distributed Algorithm

- Distributed Algorithm is a algorithm that runs on a distributed system. Each processor has its own memory and they communicate via communication networks.
- Many algorithms used in distributed system require a coordinator that performs functions needed by other processes in the system.

- **Election algorithms are designed to choose a coordinator**.

- Why Election Algorithm ?
  1. Many distributed algorithms require a process to act as a coordinator.
  2. The coordinator can be any process that organizes actions of other processes.
  3. A coordinator may fail.
  4. How is a new coordinator chosen or elected?

Election Algorithm

- **Assumptions :**
  - Each process has a unique number to distinguish them. Processes know each other's process number.

  There are two types of Election Algorithms:

  1. Bully Algorithm
  2. Ring Algorithm

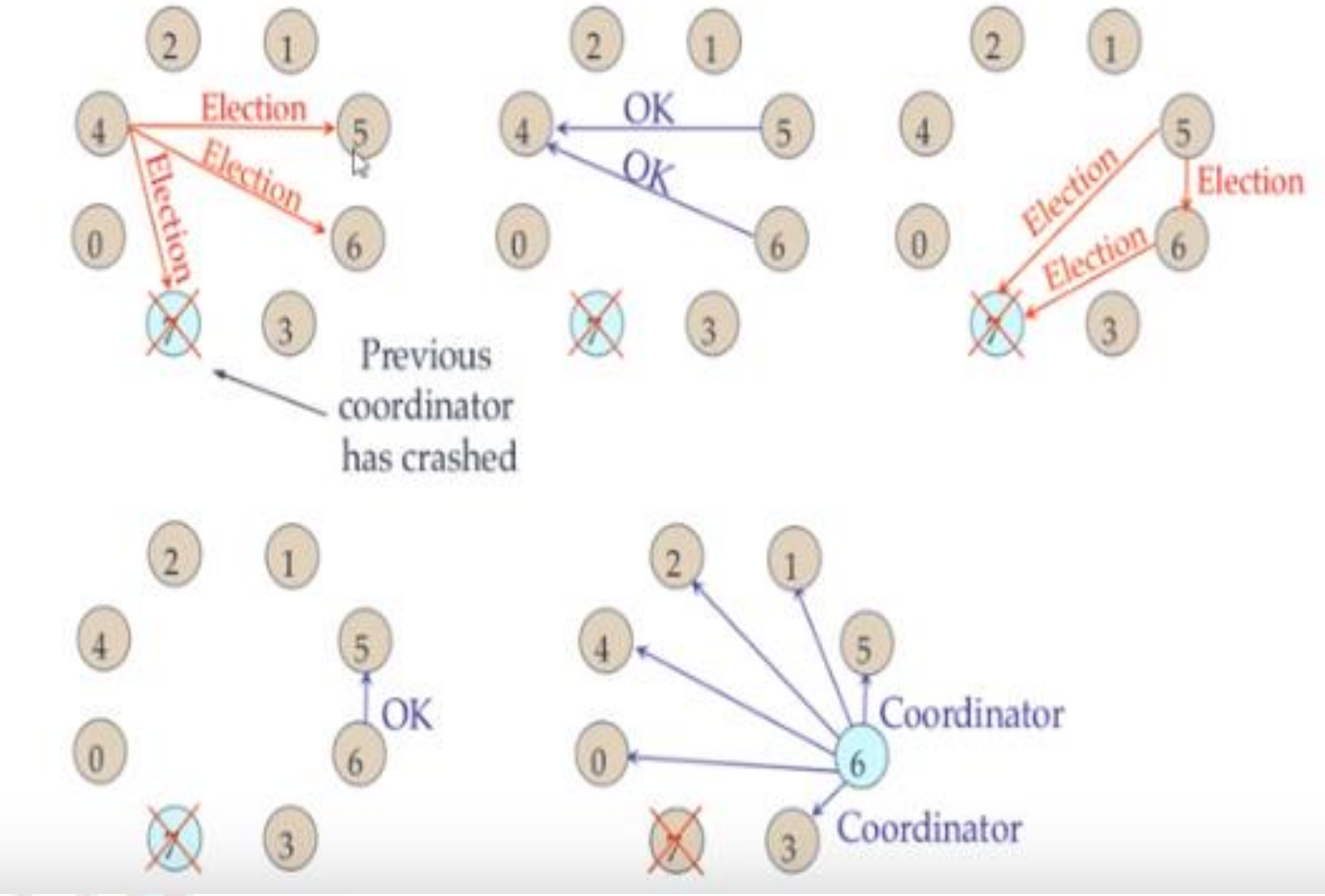
Bully Algorithm

- **Bully Algorithm :**
  This algorithm applies to system where every process can send a message to every other process in the system.

Algorithm – Suppose process P sends a message to the coordinator.

1. If coordinator does not respond to it within a time interval T, then it is assumed that coordinator has failed.
2. Now process P sends election message to every process with high priority number.
3. It waits for responses, if no one responds for time interval T then process P elects itself as a coordinator.
4. Then it sends a message to all lower priority number processes that it is elected as their new coordinator.
5. However, if an answer is received within time T from any other process Q,
   - (I) Process P again waits for time interval T' to receive another message from Q that it has been elected as coordinator.
   - (II) If Q doesn't responds within time interval T' then it is assumed to have failed and algorithm is restarted.

Bully Algorithm



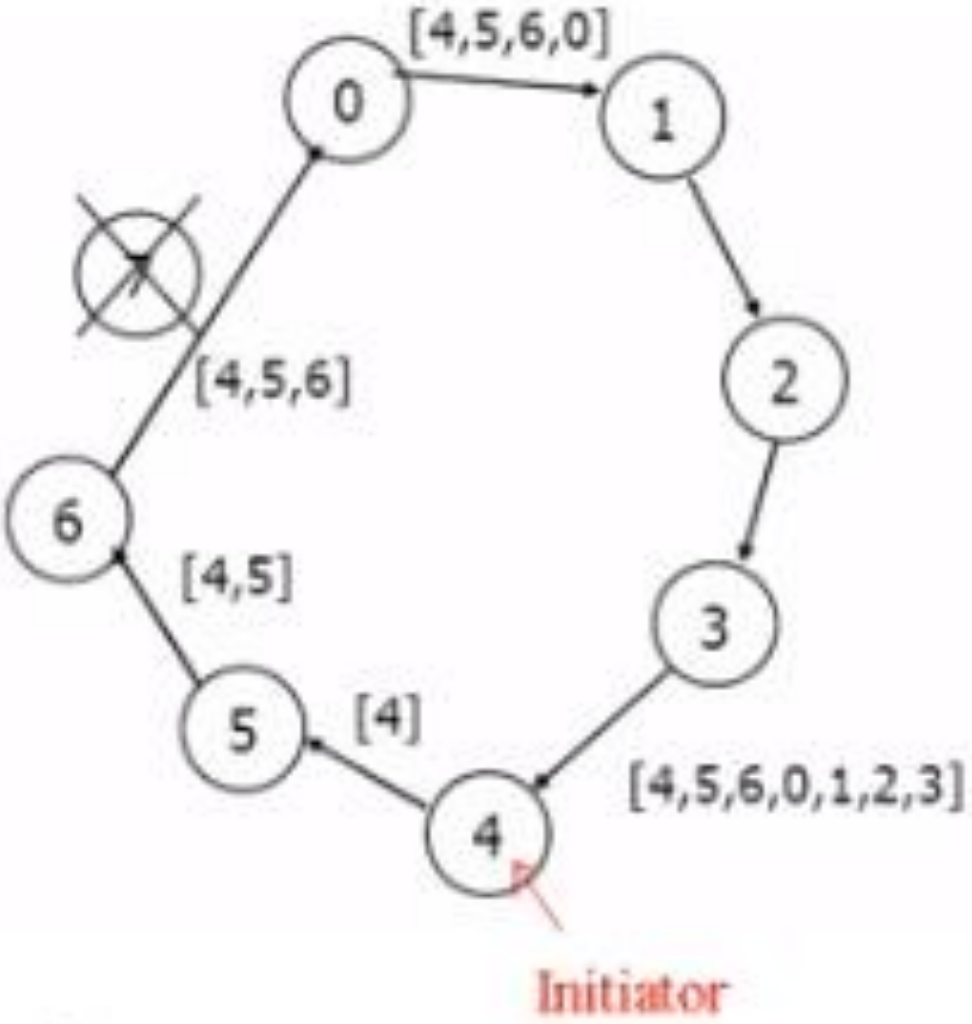
Ring Algorithm

- **Ring Algorithm :** This algorithm applies to systems organized as a ring(logically or physically). In this algorithm we assume that the link between the process are unidirectional and every process can message to the process on its right only. Data structure that this algorithm uses is active list, a list that has priority number of all active processes in the system.

Algorithm –

1. If process P1 detects a coordinator failure, it creates new active list which is empty initially. It sends election message to its neighbour on right and adds number 1 to its active list.
2. If process P2 receives message elect from processes on left, it responds in 3 ways:
   a. If message received does not contain 1 in active list then P1 adds 2 to its active list and forwards the message.
   b. If this is the first election message it has received or sent, P1 creates new active list with numbers 1 and 2. It then sends election message 1 followed by 2
   c. If Process P1 receives its own election message 1 then active list for P1 now contains numbers of all the active processes in the system. Now Process P1 detects highest priority number from list and elects it as the new coordinator.

Ring Algorithm



Implementing the solution For Ring Algorithm

1. Creating Class for Process which includes
   - i) State: Active / Inactive
   - ii) Index: Stores index of process.
   - iii) ID: Process ID

```
133 class Rr {
134
135     public int index;   // to store the index of process
136     public int id;      // to store id/name of process
137     public int f;
138     String state;       // indiactes whether active or inactive state of node
139
140 }
```

2. Import Scanner Class for getting the no of processes as an input

```
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
```

Implementing the solution For Ring Algorithm

3. Getting input from User for number of Processes and store them into object of classes.

```
23 // getting input from users
24         for (i = 0; i < num; i++) {
25             proc[i].index = i;
26             System.out.println("Enter the id of process : ");
27             proc[i].id = in.nextInt();
28             proc[i].state = "active";
29             proc[i].f = 0;
30         }
31
```

4. Sort these objects on the basis of process id.

```
33 // sorting the processes from on the basis of id
34         for (i = 0; i < num - 1; i++) {
35             for (j = 0; j < num - 1; j++) {
36                 if (proc[j].id > proc[j + 1].id) {
37                     temp = proc[j].id;
38                     proc[j].id = proc[j + 1].id;
39                     proc[j + 1].id = temp;
40                 }
41             }
42         }
43
```

Implementing the solution For Ring Algorithm

5. Make the last process id as "inactive".

```
proc[num - 1].state = "inactive";

System.out.println("\n process " + proc[num - 1].id + "select as co-ordinator");
```

6. Ask for menu 1.Election 2.Quit.

```
63          while (true) {
64              System.out.println("\n 1.election 2.quit ");
65              ch = in.nextInt();
66
```

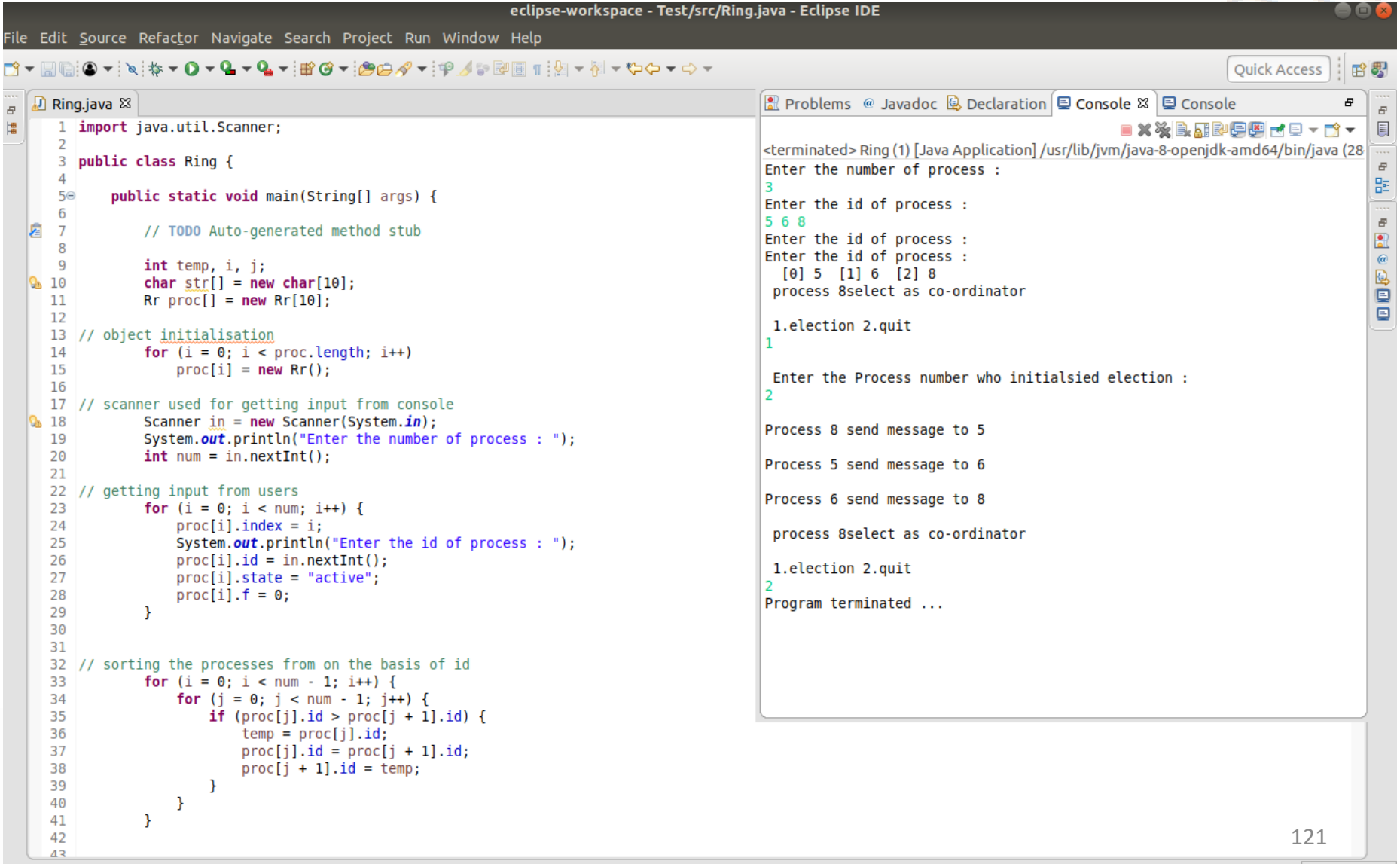
7. Ask for initializing election process.

```
switch (ch) {
case 1:
    System.out.println("\n Enter the Process number who initialsied election : ");
    init = in.nextInt();
```

8.These inputs will be used by Ring Algorithm.

Implementing the solution For Ring Algorithm

8.These inputs will be used by Ring Algorithm.




```java
import java.util.Scanner;

public class Ring {

    public static void main(String[] args) {

        // TODO Auto-generated method stub

        int temp, i, j;
        char str[] = new char[10];
        Rr proc[] = new Rr[10];

// object initialisation
        for (i = 0; i < proc.length; i++)
            proc[i] = new Rr();

// scanner used for getting input from console
        Scanner in = new Scanner(System.in);
        System.out.println("Enter the number of process : ");
        int num = in.nextInt();

// getting input from users
        for (i = 0; i < num; i++) {
            proc[i].index = i;
            System.out.println("Enter the id of process : ");
            proc[i].id = in.nextInt();
            proc[i].state = "active";
            proc[i].f = 0;
        }


// sorting the processes from on the basis of id
        for (i = 0; i < num - 1; i++) {
            for (j = 0; j < num - 1; j++) {
                if (proc[j].id > proc[j + 1].id) {
                    temp = proc[j].id;
                    proc[j].id = proc[j + 1].id;
                    proc[j + 1].id = temp;
                }
            }
        }
```

Implementation of Bully algorithm

eclipse-workspace - Test/src/Bully.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access

Bully.java

```
1
2  import java.io.InputStream;
3  import java.io.PrintStream;
4  import java.util.Scanner;
5
6  public class Bully {
7      static boolean[] state = new boolean[5];
8      int coordinator;
9
10     public static void up(int up) {
11         if (state[up - 1]) {
12             System.out.println("process" + up + "is already up");
13         } else {
14             int i;
15             Bully.state[up - 1] = true;
16             System.out.println("process " + up + "held election");
17             for (i = up; i < 5; ++i) {
18                 System.out.println("election message sent from process" + up + "to p
19             }
20             for (i = up + 1; i <= 5; ++i) {
21                 if (!state[i - 1]) continue;
22                 System.out.println("alive message send from process" + i + "to proce
23                 break;
24             }
25         }
26     }
27
28     public static void down(int down) {
29         if (!state[down - 1]) {
30             System.out.println("process " + down + "is already dowm.");
31         } else {
32             Bully.state[down - 1] = false;
33         }
34     }
35
36     public static void mess(int mess) {
37         if (state[mess - 1]) {
38             if (state[4]) {
39                 System.out.println("0K");
40             } else if (!state[4]) {
41                 int i;
42                 System.out.println("process" + mess + "election");
43                 for (i = mess; i < 5; ++i) {
```

Problems Javadoc Declaration Console Console

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
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
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
```

References

1. https://www.geeksforgeeks.org/election-algorithm-and-distributed-processing/