Assignment No - 4

Election algorithem

Election algorithem implementation :

1. Create project in eclipse

2. Create the class **Ring.java** or **Bully.java** and execute .

Ring.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;
```

```
                }
            }
        }


        for (i = 0; i < num; i++) {
            System.out.print("  [" + i + "]" + " " + proc[i].id);
        }




        int init;
        int ch;
        int temp1;
        int temp2;
        int ch1;
        int arr[] = new int[10];

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

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

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

            for (i = 0; i < num; i++) {
                proc[i].f = 0;
            }

            switch (ch) {
            case 1:
                System.out.println("\n Enter the Process number who initialsied
election : ");
                init = in.nextInt();
                temp2 = init;
                temp1 = init + 1;

                i = 0;

                while (temp2 != temp1) {
                    if ("active".equals(proc[temp1].state) && proc[temp1].f
== 0) {

                        System.out.println("\nProcess " + proc[init].id + "
send message to " + proc[temp1].id);
                        proc[temp1].f = 1;
```

```
                                        init = temp1;
                                        arr[i] = proc[temp1].id;
                                        i++;
                                }
                                if (temp1 == num) {
                                        temp1 = 0;
                                } else {
                                        temp1++;
                                }
                        }

                        System.out.println("\nProcess " + proc[init].id + " send message
to " + proc[temp1].id);
                        arr[i] = proc[temp1].id;
                        i++;
                        int max = -1;


// finding maximum for co-ordinator selection
                        for (j = 0; j < i; j++) {
                                if (max < arr[j]) {
                                        max = arr[j];
                                }
                        }

// co-ordinator is found then printing on console
                        System.out.println("\n process " + max + "select as co-
ordinator");


                        for (i = 0; i < num; i++) {

                                if (proc[i].id == max) {
                                        proc[i].state = "inactive";
                                }
                        }
                        break;
                case 2:
        System.out.println("Program terminated ...");
        return ;
                default:
                        System.out.println("\n invalid response \n");
                        break;
                }


                }
        }
```

```
}

class Rr {

        public int index;   // to store the index of process
        public int id;      // to store id/name of process
        public int f;
        String state;       // indiactes whether active or inactive state of node

}
```

Bully.java:-

```
import java.io.InputStream;
import java.io.PrintStream;
import java.util.Scanner;

public class Bully {
   static boolean[] state = new boolean[5];
   int coordinator;

   public static void up(int up) {
      if (state[up - 1]) {
         System.out.println("process" + up + "is already up");
      } else {
         int i;
         Bully.state[up - 1] = true;
         System.out.println("process " + up + "held election");
         for (i = up; i < 5; ++i) {
            System.out.println("election message sent from process" + up + "to process" + (i
+ 1));
         }
         for (i = up + 1; i <= 5; ++i) {
            if (!state[i - 1]) continue;
            System.out.println("alive message send from process" + i + "to process" + up);
            break;
         }
      }
   }

   public static void down(int down) {
      if (!state[down - 1]) {
         System.out.println("process " + down + "is already dowm.");
      } else {
         Bully.state[down - 1] = false;
      }
   }
```

```
    public static void mess(int mess) {
        if (state[mess - 1]) {
            if (state[4]) {
                System.out.println("0K");
            } else if (!state[4]) {
                int i;
                System.out.println("process" + mess + "election");
                for (i = mess; i < 5; ++i) {
                    System.out.println("election send from process" + mess + "to process " + (i +
1));
                }
                for (i = 5; i >= mess; --i) {
                    if (!state[i - 1]) continue;
                    System.out.println("Coordinator message send from process" + i + "to all");
                    break;
                }
            }
        } else {
            System.out.println("Prccess" + mess + "is down");
        }
    }

    public static void main(String[] args) {
        int choice;
        Scanner sc = new Scanner(System.in);
        for (int i = 0; i < 5; ++i) {
            Bully.state[i] = true;
        }
        System.out.println("5 active process are:");
        System.out.println("Process up  = p1 p2 p3 p4 p5");
        System.out.println("Process 5 is coordinator");
        do {
            System.out.println(".....");
            System.out.println("1 up a process.");
            System.out.println("2.down a process");
            System.out.println("3 send a message");
            System.out.println("4.Exit");
            choice = sc.nextInt();
            switch (choice) {
                case 1: {
                    System.out.println("bring proces up");
                    int up = sc.nextInt();
                    if (up == 5) {
                        System.out.println("process 5 is co-ordinator");
                        Bully.state[4] = true;
                        break;
                    }
```

```
                Bully.up(up);
                break;
            }
            case 2: {
                System.out.println("bring down any process.");
                int down = sc.nextInt();
                Bully.down(down);
                break;
            }
            case 3: {
                System.out.println("which process will send message");
                int mess = sc.nextInt();
                Bully.mess(mess);
            }
        }
    } while (choice != 4);
  }
}
```