**Code**

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

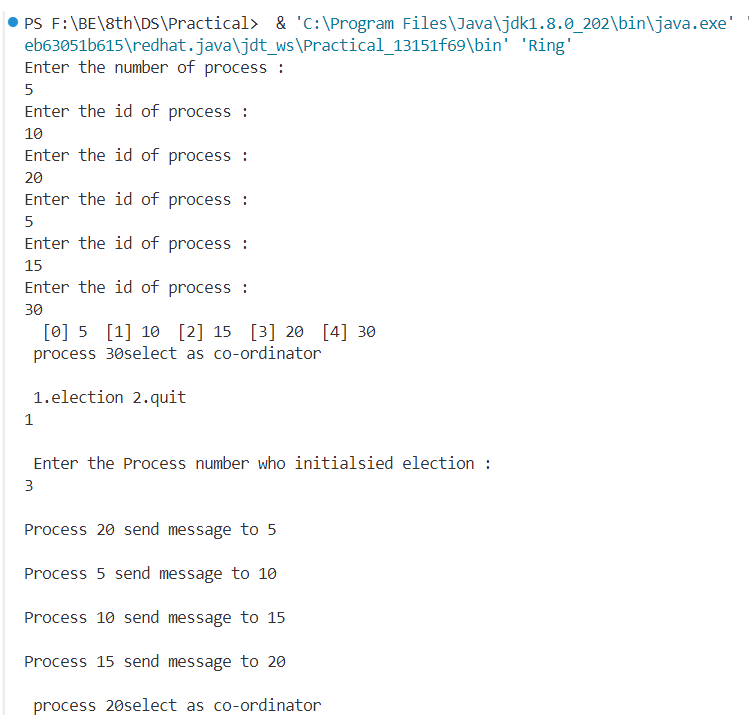
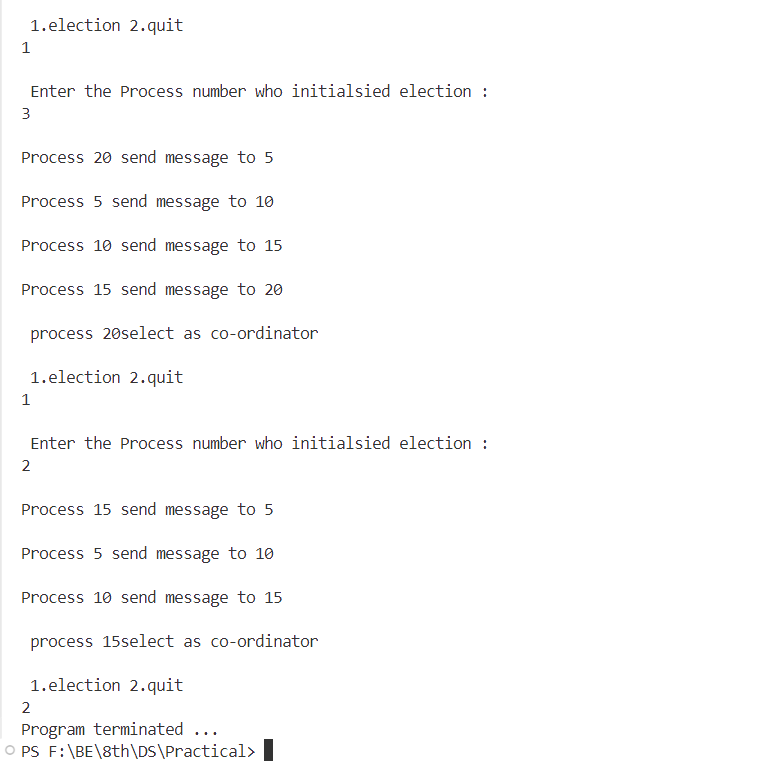
    public int id;      // to store id/name of process

    public int f;

    String state;       // indiactes whether active or inactive state of node

}

**Output**

** **