Lab 1: Implementation of Election Algorithm: The Ring Algorithm

An algorithm for choosing a unique process to play a particular role is called an election algorithm. For example, in a variant of the central-server algorithm for mutual exclusion, the 'server' is chosen from among the processes pi, (i = 1, 2,... N)that need to use the critical section. An election algorithm is needed for this choice. It is essential that all the processes agree on the choice. Afterwards, if the process that plays the role of server wishes to retire then another election is required to choose a replacement.

We say that a process calls the election if it takes an action that initiates a particular run of the election algorithm. An individual process does not call more than one election at a time, but in principle the N processes could call N concurrent elections. An important requirement is for the choice of elected process to be unique, even if several processes call elections concurrently. For example, two processes could decide independently that a coordinator process has failed, and both call elections.

Without loss of generality, we require that the elected process be chosen as the one with the largest identifier. The 'identifier' may be any useful value, as long as the identifiers are unique and totally ordered.

Ring Algorithm

- Any process p_i that discovers the old coordinator has failed initiates an "Election" message that contains p_i's own id:attr. This is the *initiator* of the election.
- When a process p_i receives an "Election" message, it compares the attr in the message with its own attr.
    - If the arrived attr is greater, p_i forwards the message.
    - If the arrived attr is smaller and p_i has not forwarded an election message earlier, it overwrites the message with its own id:attr, and forwards it.
    - If the arrived id:attr matches that of p_i, then p_i' s attr must be the greatest (why?), and it becomes the new coordinator. This process then sends an "Elected" message to its neighbor with its id, announcing the election result.
- When a process p_i receives an "Elected" message, it
    - sets its variable $elected_i \leftarrow$ id of the message.
    - forwards the message unless it is the new coordinator.

Source Code

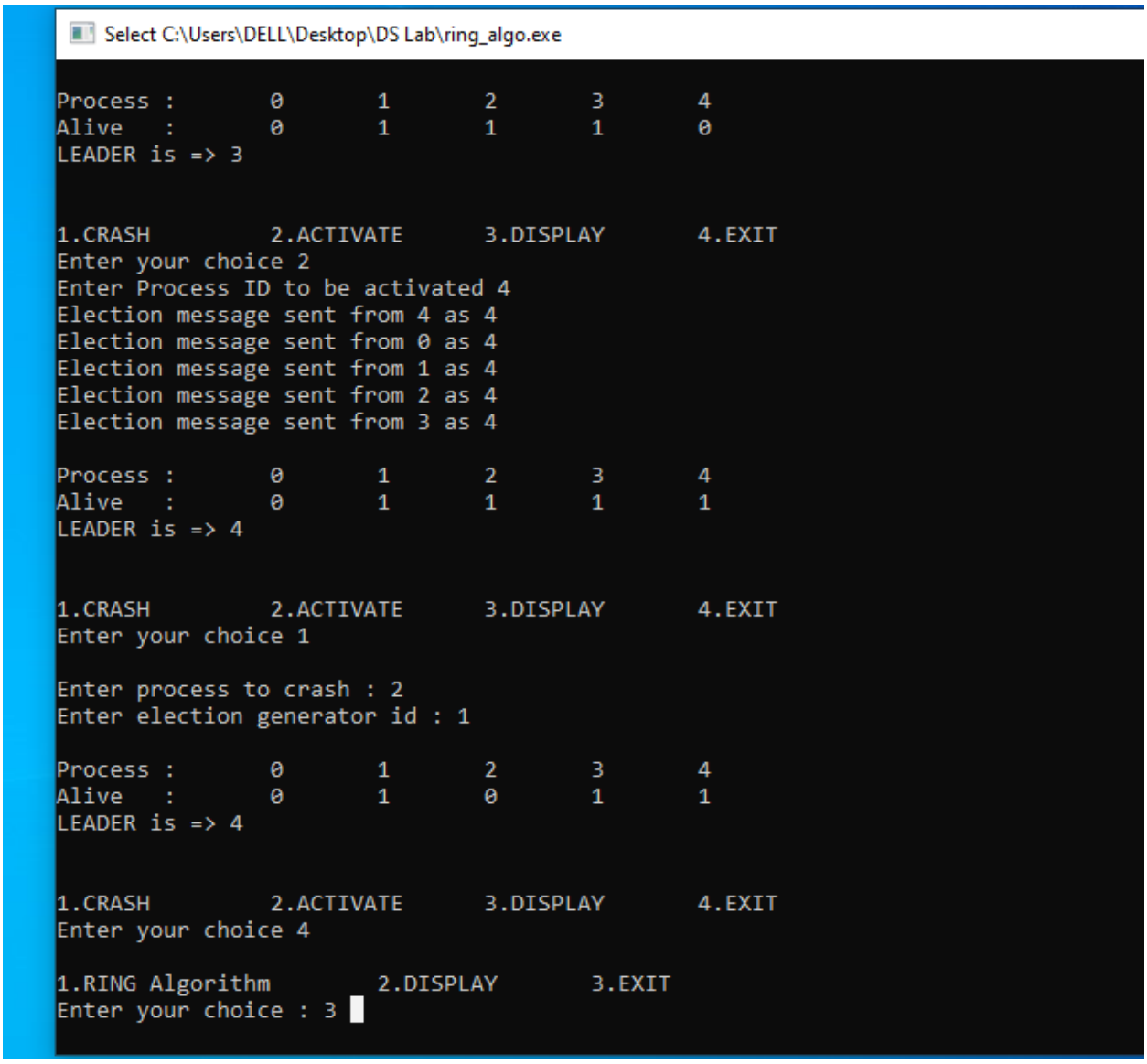
```
#include <iostream>
using namespace std;
#define MAX 20

int process[MAX], n, leader;
void display();
void ring();

int main()
{
    int fchoice, i;
    cout << "Enter no. of processes : ";
    cin >> n;
    cout << endl
         << "Enter processes state below, i.e alive or dead (1/0)" << endl;
    for (i = 0; i < n; i++)
    {
        cout << "for Process " << i << " : ";
        cin >> process[i];
        if (process[i])
            leader = i;
    }

    display();
    do
    {
        cout << "\n1.RING Algorithm \t2.DISPLAY \t3.EXIT\n";
        cout << "Enter your choice : ";
        cin >> fchoice;
        switch (fchoice)
        {
        case 1:
            ring();
            break;

        case 2:
            display();
            break;

        case 3:
            break;

        default:
            break;
        }
    } while (fchoice != 3);

    return 0;
}
```

```
void display()
{
    int i;
    cout << "\nProcess :  ";
    for (i = 0; i < n; i++)
        cout << "\t" << i;

    cout << "\nAlive   :  ";
    for (i = 0; i < n; i++)
        cout << "\t" << process[i];
    cout << endl
         << "LEADER is => " << leader << endl
         << endl;
}


// Implementing RING Algorithm
=======================================================
void ring()
{
    int schoice, crash, activate, i, gid, flag, subLeader;
    do
    {
        cout << "\n1.CRASH \t2.ACTIVATE \t3.DISPLAY \t4.EXIT\n";
        cout << "Enter your choice ";
        cin >> schoice;
        switch (schoice)
        {
        case 1:
            cout << "\nEnter process to crash : ";
            cin >> crash;

            if (process[crash])
                process[crash] = 0;
            else
                cout << "Process " << crash << " is already dead!" << endl;
            do
            {
                cout << "Enter election generator id : ";
                cin >> gid;
                if (gid == leader)
                    cout << "Not a valid generator id!!" << endl;
            } while (gid == leader);

            if (crash == leader)
            {
                subLeader = 0;
                for (i = 0; i < n; i++)
                {
                    int index = (i + gid) % n;
                    if (process[index] && subLeader < index)
                    {
```

```
                        subLeader = index;
                    }
                    cout << "Election message sent from " << index << " as " <<
subLeader << endl;
                }

                leader = subLeader;
            }
            display();
            break;

        case 2:
            cout << "Enter Process ID to be activated ";
            cin >> activate;
            if (!process[activate])
                process[activate] = 1;
            else
            {
                cout << "Process " << activate << " is already alive!" << endl;
                break;
            }

            subLeader = activate;
            for (i = 0; i < n; i++)
            {
                int index = (i + activate) % n;
                if (process[index] && subLeader < index)
                {
                    subLeader = index;
                }
                cout << "Election message sent from " << index << " as " <<
subLeader << endl;
            }

            leader = subLeader;
            display();
            break;

        case 3:
            display();
            break;

        default:
            break;
        }
    } while (schoice != 4);
}
```

Output

```
Select C:\Users\DELL\Desktop\DS Lab\ring_algo.exe
Enter no. of processes : 5

Enter processes state below, i.e alive or dead (1/0)
for Process 0 : 0
for Process 1 : 1
for Process 2 : 1
for Process 3 : 1
for Process 4 : 1

Process :       0       1       2       3       4
Alive   :       0       1       1       1       1
LEADER is => 4


1.RING Algorithm        2.DISPLAY       3.EXIT
Enter your choice : 1

1.CRASH         2.ACTIVATE      3.DISPLAY       4.EXIT
Enter your choice 3

Process :       0       1       2       3       4
Alive   :       0       1       1       1       1
LEADER is => 4


1.CRASH         2.ACTIVATE      3.DISPLAY       4.EXIT
Enter your choice 1

Enter process to crash : 4
Enter election generator id : 3
Election message sent from 3 as 3
Election message sent from 4 as 3
Election message sent from 0 as 3
Election message sent from 1 as 3
Election message sent from 2 as 3

Process :       0       1       2       3       4
Alive   :       0       1       1       1       0
LEADER is => 3


1.CRASH         2.ACTIVATE      3.DISPLAY       4.EXIT
```


Select C:\Users\DELL\Desktop\DS Lab\ring_algo.exe
Process : 0 1 2 3 4
Alive   : 0 1 1 1 0
LEADER is => 3
1.CRASH 2.ACTIVATE 3.DISPLAY 4.EXIT
Enter your choice 2
Enter Process ID to be activated 4
Election message sent from 4 as 4
Election message sent from 0 as 4
Election message sent from 1 as 4
Election message sent from 2 as 4
Election message sent from 3 as 4
Process : 0 1 2 3 4
Alive   : 0 1 1 1 1
LEADER is => 4
1.CRASH 2.ACTIVATE 3.DISPLAY 4.EXIT
Enter your choice 1
Enter process to crash : 2
Enter election generator id : 1
Process : 0 1 2 3 4
Alive   : 0 1 0 1 1
LEADER is => 4
1.CRASH 2.ACTIVATE 3.DISPLAY 4.EXIT
Enter your choice 4
1.RING Algorithm 2.DISPLAY 3.EXIT
Enter your choice : 3