Nicole Serna

Zachary Serna

3/18/2022

Professor Doina Bein

CPSC 479

Project 1: Electing Two Leaders in a Ring Topology

Group Members:

Zach Serna: serna648@csu.fullerton.edu

Nicole Serna: NicoleSerna@csu.fullerton.edu

Language Used: C++

How to Compile:

1. add the file to your directory
2. compile the .cpp file using mpic++ (filename) -o ./(desiredexecname)
3. run the exe with mpirun -n (desired number of processors) ./(desiredexecname)

Description: Our implementation of the Electing Two Leaders in a Ring Topology assignment. The program will generate N identical copies of nodes with a 5 digit ID attached to these nodes.

The nodes' random ID's are generated as follows:

The first two digits NN is a randomly generated number by the process; the process generates a random number in the interval 10..99 and: If the value is negative, then take the absolute value of it If the value is less than 10, then add 10 to it If the value is greater than 100, then take modulo 100 OR find a formula to make it exactly in the range 10..99 The next two digits RR represent the rank. Since the number of processes in the ring must be greater than 5, if the rank is one digit (e.g. rank 5) then left-pad it with one 0’s (.e.g 05). The last digit D represents the mod 2 of the random value generated earlier NN.

After being generated, the nodes will then determine the PRESIDENT and the VICE PRESIDENT. The president is the smallest even value, while the vice president is the largest odd value, by default, the values for president and vice president are generated to 99998 and 1 respectively. The president and vice presidents will then be determined by traveling across the ring, ending at Rank 0 to determine their final rank. The newly elected president and vice president as well as their respective rank is then printed to the screen.

Implementation: We chose to implement the Concurrent two leader election algorithm in our project. In this version of the program, an array will store both the values of the president and the Vice President. These values will be checked for simultaneously as we traverse through the ring, as opposed to once for the president and once for the vice president.

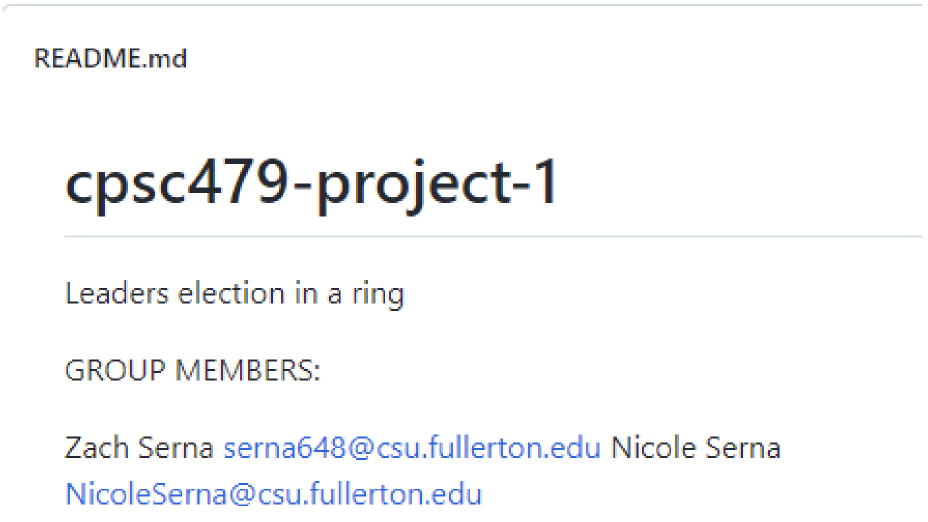
Pseudocode:

```
//Project 479 PseudoCode

//Main

//variables. . . .

//int rank
//int size
//int ID to hold an integer version of the ID NNRRD
//int NN for the randomly generated number portion of the ID, NN

//char arrayID[5] to hold char version ID, needed to build and connect all aspects of ID
//char NNArray[2] to hold NN values converted to char
//char RRArray[2] to hold RR (rank) values as a char
//char Dvalue[1] to hold result of taking mod 2 of NN as a Char

//int array PRVR[4] (0:President Value, 1:President Rank, 2:VP Value, 3:VP Rank)


// Implementation
//Initialize MPI
//Initialize Rank and Size
//seed the random number generator
```

```
//check if rank is 0
    //if rank is 0
    //generate random number 10-99, NN
    // combine random number with rank (RR), 00 in this case
    // perform mod 2 operation on NN, result D
    // concatenate NN RR and D into one int, ID
    // if ID is even
        //PRVR[0] = ID
        //PRVR[1] = rank
        //PRVR[2] = 1
        //PRVR[3] = rank
    // else ID is odd
        //PRVR[0] = 99998
        //PRVR[1] = rank
        //PRVR[2] = ID
        //PRVR[3] = rank
    //MPI SEND sends PRVR array (current President and VP information)
    //RECV PRVR array (from last process in rank)
    //Print out President, President Rank, VP, and VP Rank

//else, rank is  NOT zero

    //RECV PRVR array (updated pres and vp from previous rank)
    //generate random number 10-99, NN
    // combine random number with rank (RR), 00 in this case
    // perform mod 2 operation on NN, result D
    // concatenate NN RR and D into one int, ID
    //if ID is even
        //if ID < PRVR[0]
            //PRVR[0] = ID
            //PRVR[1] = rank
        //else nothing
    //if ID is odd
        //if ID >PRVR[2]
            //PRVR[2] = ID
            //PRVR[3] = rank
    //MPI SEND sends PRVR array (current President and VP information)

//MPI finalize
//Done@!
```

Screenshots:

Group Members:



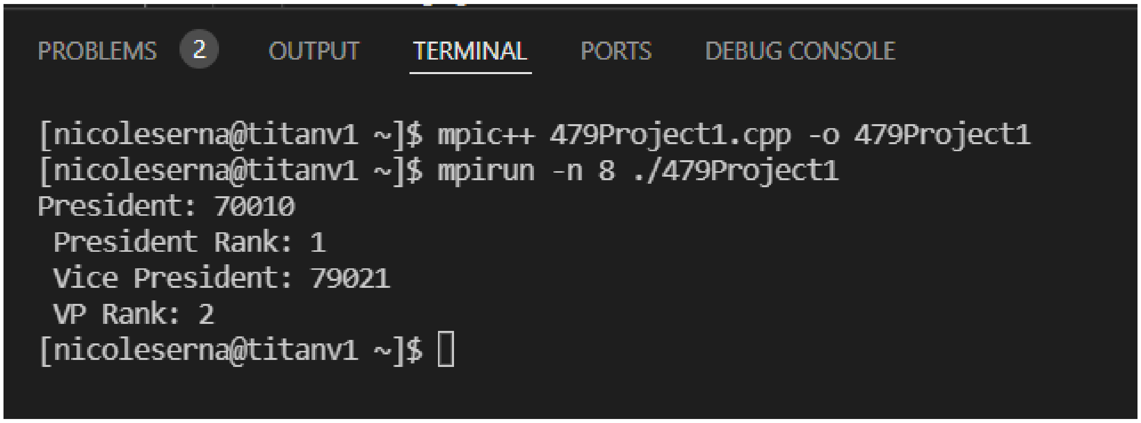
cpsc479-project-1

Leaders election in a ring

GROUP MEMBERS:

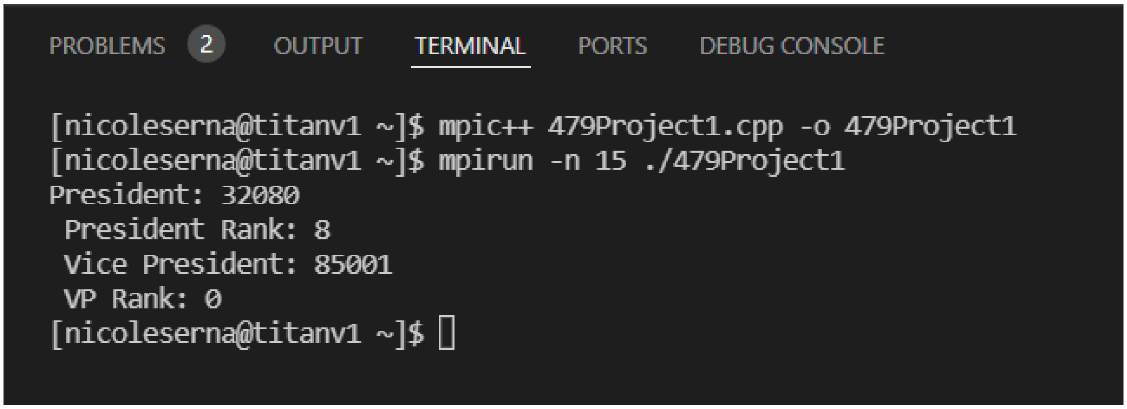
Zach Serna serna648@csu.fullerton.edu Nicole Serna NicoleSerna@csu.fullerton.edu

Test Run 1:



```
[nicoleserna@titanv1 ~]$ mpic++ 479Project1.cpp -o 479Project1
[nicoleserna@titanv1 ~]$ mpirun -n 8 ./479Project1
President: 70010
 President Rank: 1
 Vice President: 79021
 VP Rank: 2
[nicoleserna@titanv1 ~]$
```

Test Run 2:



```
[nicoleserna@titanv1 ~]$ mpic++ 479Project1.cpp -o 479Project1
[nicoleserna@titanv1 ~]$ mpirun -n 15 ./479Project1
President: 32080
 President Rank: 8
 Vice President: 85001
 VP Rank: 0
[nicoleserna@titanv1 ~]$
```