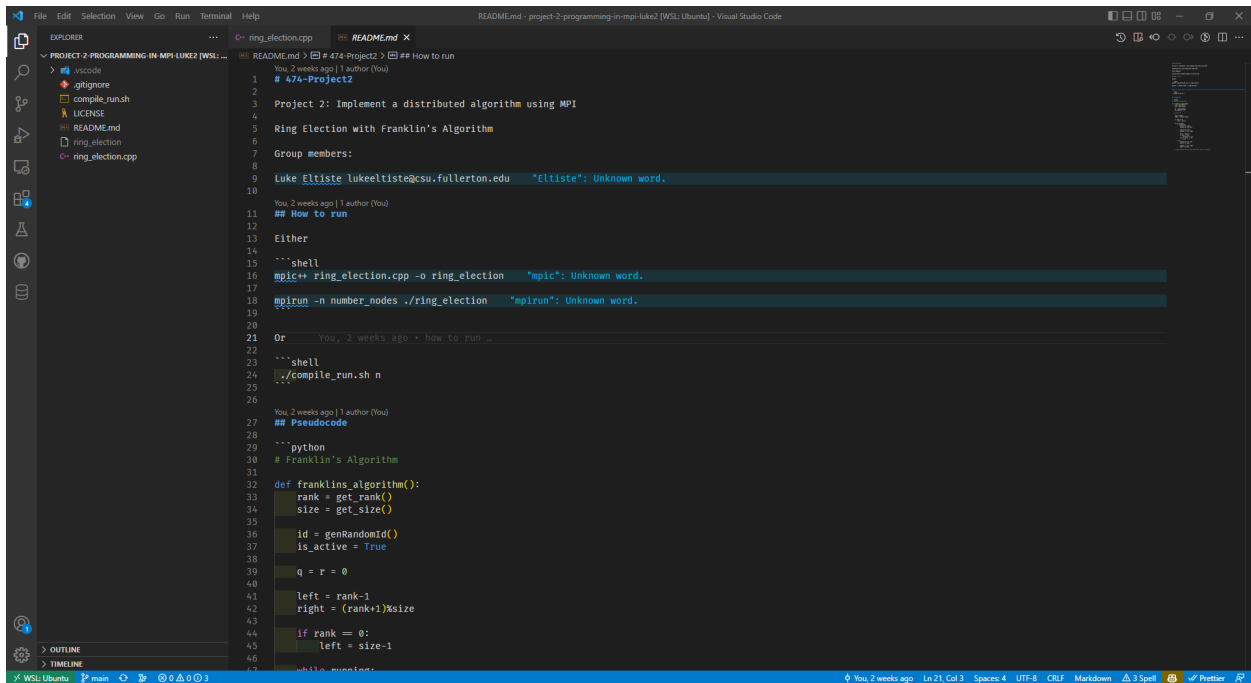


Project 2 - Franklin's Election Algorithm

lukeeltiste@csu.fullerton.edu - Luke Eltiste



```
1 # 474-Project2
2
3 Project 2: Implement a distributed algorithm using MPI
4
5 Ring Election with Franklin's Algorithm
6
7 Group members:
8
9 Luke Eltiste lukeeltiste@csu.fullerton.edu "eltiste": Unknown word.
10
11 You, 2 weeks ago | 1 author (You)
12 ## How to run
13
14 Either
15
16 ```shell
17 mpic++ ring_election.cpp -o ring_election "mpic": Unknown word.
18
19 mpiexec -n number_nodes ./ring_election "mpirun": Unknown word.
20
21 Or
22
23 ```shell
24 ./compile_run.sh n
25
26
27 You, 2 weeks ago | 1 author (You)
28 ## Pseudocode
29
30 ```python
31 # Franklin's Algorithm
32
33 def franklins_algorithm():
34     rank = get_rank()
35     size = get_size()
36     id = genRandomId()
37     is_active = True
38
39     q = r = 0
40
41     left = rank-1
42     right = (rank+1)%size
43
44     if rank == 0:
45         left = size-1
46
47     while True:
48         if is_active:
49             send id to right
50             receive q from left
51
52             send id to left
53             receive r from right
```

Pseudocode

```
def franklins_algorithm():
    rank = get_rank()
    size = get_size()
    id = genRandomId()
    is_active = True
    q = r = 0
    left = rank-1
    right = (rank+1)%size
    if rank == 0:
        left = size-1

    while True:
        if is_active:
            send id to right
            receive q from left

            send id to left
            receive r from right
```

```

        max_v = max(q,r)
        if max_v > id:
            is_active = False
        else if max_v == id:
            break
    else:
        receive q from left
        send q to right

        receive r from right
        send r to left

# leader breaks out of loop and kills rest of processes

```

How to run the code

I've created a simple shell script in the repo named "compile_run.sh" which can be called like `./compile_run.sh 10` replace '10' with the number of nodes you would like to have in the election

If you would like to run the commands manually without the shell script you would use the commands `"mpic++ ring_election.cpp -o ring_election"` and then `"mpirun -n 10 ./ring_election"` replace '10' with the number of nodes you want to use.

Example Output

```

└─ ./dev/school/CPSC 474 PDC/project-2-programming-in-mpi-luke2 on p main +1 -1 [!?]
└─ ./compile_run.sh 10
1 is now passive with id: 846930886
2 is now passive with id: 1681692777
3 is now passive with id: 1714636915
4 continues to next round
4 continues to next round
7 continues to next round
7 is now passive with id: 1649760492
8 is now passive with id: 596516649
0 continues to next round
0 is now passive with id: 1804289383
5 is now passive with id: 424238335
6 is now passive with id: 719885386
9 is now passive with id: 1189641421
4 is now the leader with id: 1957747793
Abort(0) on node 4 (rank 4 in comm 0): application called MPI_Abort(MPI_COMM_WORLD, 0) - process 4

```

```
.../dev/school/CPSC 474 PDC/project-2-programming-in-mpi-luke2 on  main  
→ ./compile_run.sh 5  
0 is now passive with id: 1804289383  
1 is now passive with id: 846930886  
2 is now passive with id: 1681692777  
3 is now passive with id: 1714636915  
4 continues to next round  
4 is now the leader with id: 1957747793  
Abort(0) on node 4 (rank 4 in comm 0): application called MPI_Abort(MPI_COMM_WORLD, 0) - process 4
```