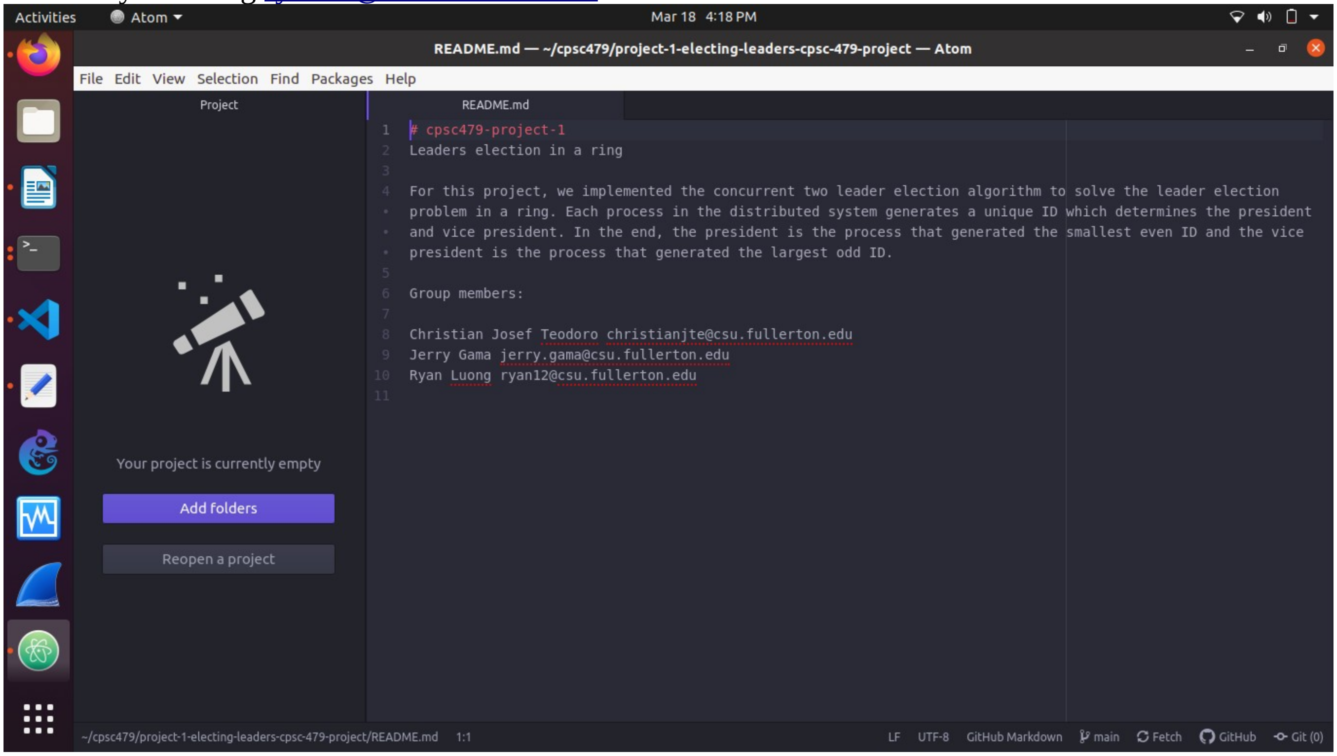
Project 1

Group Members:

Christian Josef Teodoro christianjte@csu.fullerton.edu
Jerry Gama jerry.gama@csu.fullerton.edu
Ryan Luong ryan12@csu.fullerton.edu


Activities
Atom
Mar 18 4:18 PM
README.md — ~/cpsc479/project-1-electing-leaders-cpsc-479-project — Atom
File Edit View Selection Find Packages Help
Project
README.md
Your project is currently empty
Add folders
Reopen a project
~/cpsc479/project-1-electing-leaders-cpsc-479-project/README.md 1:1
LF UTF-8 GitHub Markdown main Fetch GitHub Git (0)

```
# cpsc479-project-1
Leaders election in a ring

For this project, we implemented the concurrent two leader election algorithm to solve the leader election
problem in a ring. Each process in the distributed system generates a unique ID which determines the president
and vice president. In the end, the president is the process that generated the smallest even ID and the vice
president is the process that generated the largest odd ID.

Group members:

Christian Josef Teodoro christianjte@csu.fullerton.edu
Jerry Gama jerry.gama@csu.fullerton.edu
Ryan Luong ryan12@csu.fullerton.edu
```

Summary:

For this project, we implemented the concurrent two leader election algorithm to solve the leader election problem in a ring. Each process in the distributed system generates a unique ID which determines the president and vice president. In the end, the president is the process that generated the smallest even ID and the vice president is the process that generated the largest odd ID. This report includes the pseudo code of the algorithm implemented for the project and two examples of the output.

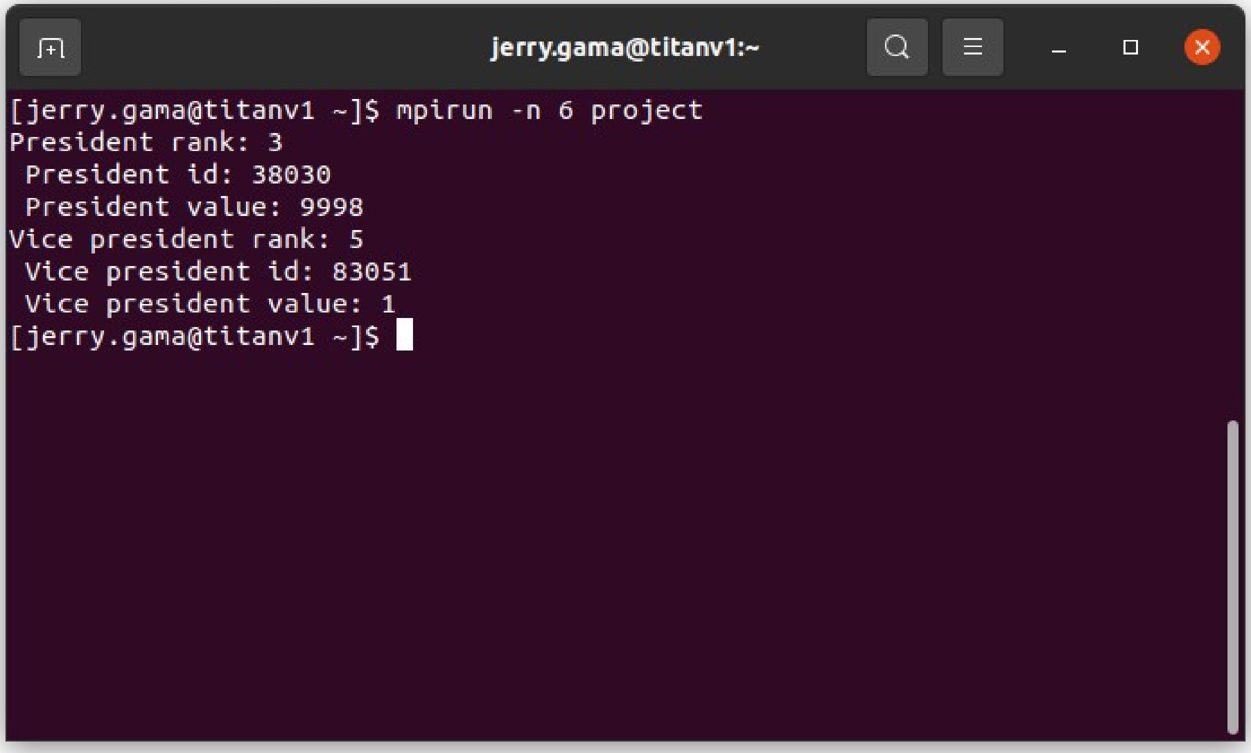
Pseudocode:

```
MPI_INIT
MPI_RANK
MPI_SIZE
MPI_BARRIER
presidentAndVicePresident[4] = [president rank, president id (even), vice president rank, vice
                                    president id (odd)]

random = number between 10 and 99
NN = random * 1000; //Get a five digit place number
RR = rank * 10; //Get RR to the hundredth place
D = random % 2;
id = NN + RR + D;
```

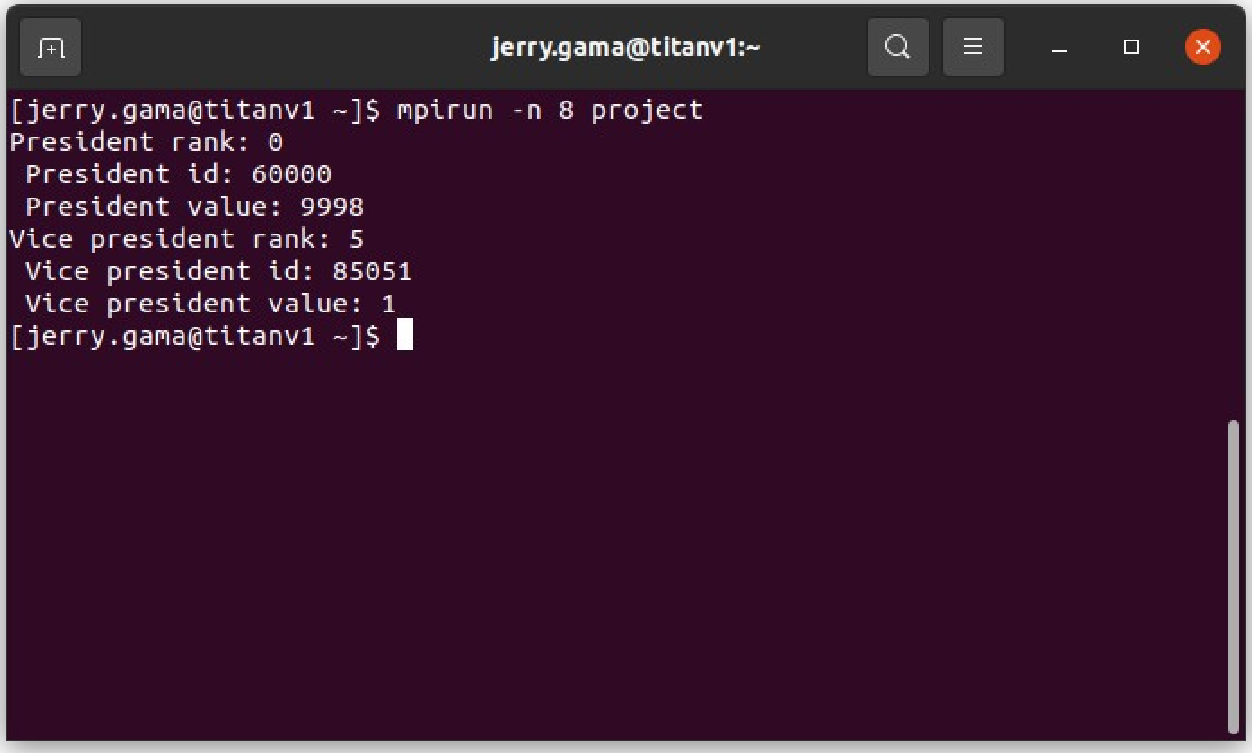
```
if rank is 0:
  if id % 2 is even:
    presidentAndVicePresident[4] = [0, id, 0, 0]
  else:
    presidentAndVicePresident[4] = {0, 0, 0, id]
else:
  receive presidentAndVicePresident from process rank - 1

send presidentAndVicePresident to the next process (rank + 1)

if id % 2 is 0 and id is less than or equal to president id:
  presidentAndVicePresident[0] = rank
  presidentAndVicePresident[1] = id
else if id % 2 is odd and id is greater than or equal to vice president id:
  presidentAndVicePresident[2] = rank
  presidentAndVicePresident[3] = id

if rank is 0:
  receive presidentAndVicePresident from rank - 1
  print the president rank, id, and value
  print the vice president rank, id, and value

MPI_FINALIZE
end
```

The code was compiled and run using:

```
mpic++ -o project project.cpp
mpirun -n 6 project
```



```
[jerry.gama@titanv1 ~]$ mpirun -n 6 project
President rank: 3
 President id: 38030
 President value: 9998
Vice president rank: 5
 Vice president id: 83051
 Vice president value: 1
[jerry.gama@titanv1 ~]$
```



```
[jerry.gama@titanv1 ~]$ mpirun -n 8 project
President rank: 0
 President id: 60000
 President value: 9998
Vice president rank: 5
 Vice president id: 85051
 Vice president value: 1
[jerry.gama@titanv1 ~]$
```