Furkan Tuna 69730
Yiğithan Şanlık 64117

COMP 304 - PROJECT 2
Los Shellos Hermanos

PART1

In part 1, we have created functions in order to handle events of voting simulation. The **cast_vote** function uses a random number to decide which candidate the voter votes for, based on the given probabilities. Inside **voter_process**, voters wait for their **queue_number** to match **next_voter** is incremented, allowing the next voter in the queue to vote. In **voter_process**, we have also ensured that only one voter can be in the station at a time. The **generate_voters** function generates a new voter at each second with a probability **'p'** and then sleeps. Our simulation uses the **time()** function inside the **generate_voters** function to continuously check whether the difference between the start time and current time exceeded the simulation time. The **polling_station_mutex** ensures that only one voter votes at a time. **next_voter_cond** signals waiting voters when the current voter finishes voting.

We compile our program with:

g++ -o electronic_voting main.cpp sleep.cpp -lpthread

After that we run our program with:

./electronic_voting -t 200 -p 0.5

Output:

```
electronic_voting  main.cpp  README.md  sleep.cpp  sleep.hh
root@server:~/Desktop/project-2-los-shellos-hermanos# ./electronic_voting -t 60
-p 0.5
 Votes for Mary: 14
Votes for John: 4
Votes for Anna: 13
root@server:~/Desktop/project-2-los-shellos-hermanos#  ./electronic_voting -t 60
-p 0.5
Votes for Mary: 16
Votes for John: 4
Votes for Anna: 11
root@server:~/Desktop/project-2-los-shellos-hermanos# ./electronic_voting -t 200
 -p 0.5
Votes for Mary: 39
Votes for John: 20
Votes for Anna: 42
```

PART2

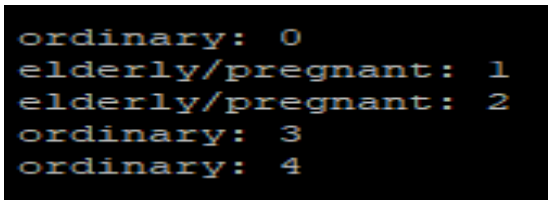
For this part, we have added a priority system for voters. The queueing machine favors elderly and pregnant voters, which creates with '**1-p'** probability at time **'t'**. The voting system allows elderly or pregnant voters to vote until one of two conditions is met:

1. No more elderly or pregnant women are waiting.
2. Five or more non-elderly non-pregnant voters are lined up to vote.

This system could potentially lead to starvation of elderly and pregnant voters because if there are always 5 or more non-elderly non-pregnant voters waiting in line, the elderly or pregnant voters may never get a chance to vote.
This problem is solved in the code by checking the count of consecutive non-priority voters that have voted. When five non-priority voters have voted consecutively, the system ensures that the next voter is a priority voter, even if it means a non-priority voter has to wait. We created a voterInfo struct in order to hold information about voters. In **voter_process**, the voter waits until it's their turn and no more than five non-priority voters have voted consecutively before them. **non_priority_wait_count** keeps track of the number of non-priority voters that have voted consecutively. If the current voter is a priority voter, **non_priority_wait_count** is reset to zero. If the current voter is a non-priority voter, **non_priority_wait_count** is incremented. When **non_priority_wait_count** reaches 5, it is reset to zero to ensure the next voter is a priority voter. In the generate_voters function, we have created a new voter every second. The type of voter (priority or non-priority) is decided based on the generated random probability. If the generated random probability is greater than p, the voter is a priority voter.

We run our program with ./electronic_voting -t 200 -p 0.5

SS:



PART3

We run our program with ./electronic_voting -t 200 -p 0.5 -f 0.1

Explanation:

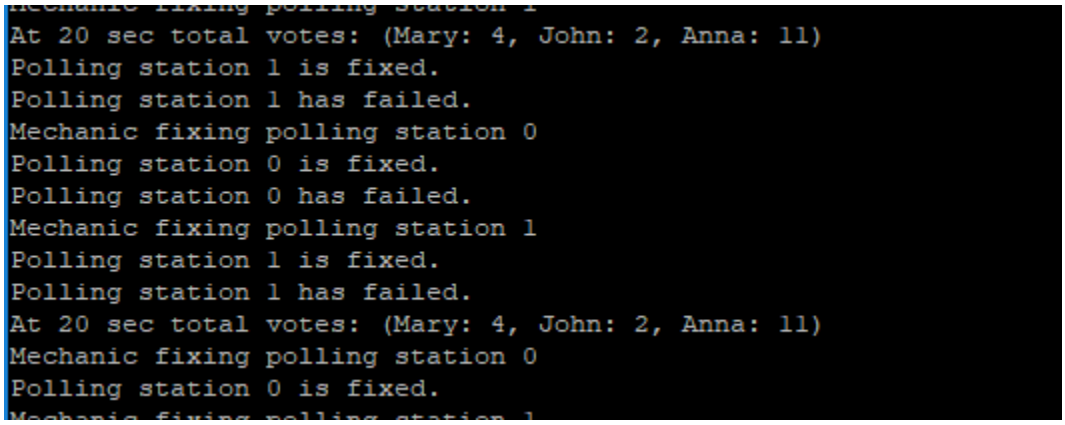
In this part, we implemented a failure situation in our **electronic_voting** simulation. We assumed that every 10t seconds, the polling station fails with probability f and voters have to wait until the polling station is fixed.

We added case ‘f’ in our switch case inside of our main function.

Also, we created “**mechanic_process**” which simulates the failure of the polling station every 10 seconds.We have created a **‘failure_probability’** variable for that. It checks if a polling station has failed and repairs it if necessary. We also modified our “**voter_process**” function. Now, it has an ability to wait if the polling station has failed.In the **generate_voters** function, a counter **failure_check_count0er** has been added to check for potential failure every 10t seconds. When this counter reaches 10, a random probability is generated. If this probability is less than **failure_probability**, the polling station is considered to have failed. The polling station is then "repaired" by sleeping for 5t seconds, after which **polling_station_failed** is set to false again.

Output:

Here we can see that when we set -f 0.5, our election systems often fail and after that the mechanic fixes it and it continues.



PART4

Our main purpose of PART IV was, instead of having a single polling station, create multiple polling stations with assigned “-c” buttons. We assign new voters to the polling station which is least busy. Basically, we initialize multiple polling stations inside of our main function. We have created a struct for polling stations since there will be plenty of polling stations for a voting simulation.

After that, we created mechanic threads for each polling station. Also, we implemented it like, each polling station has its potential to fail (implemented it on Part III) and therefore requires a mechanic thread to fix it.

After this, we implemented generation of Voters and Station Allocations: when a voter is generated, they are allocated to the polling station with the smallest que as a requirement in our PDF.

In the **voter_process** function, every voter casts their vote at the designated polling place. Each polling place's voting counters are independent since the votes for the candidates at each station are recorded individually. The unique polling station ID is also generated for the log data.

At the end of our electronic_voting simulation, mutex gets destroyed.

We run our program with ./electronic_voting -t 200 -p 0.5 -f 0.1 -c 3

Output:

```
At 1.68469e+09 sec, polling station 0, elderly/pregnant: 17
At 20 sec total votes: (Mary: 11, John: 6, Anna: 23)
At 1.68469e+09 sec, polling station 1, elderly/pregnant: 14
At 1.68469e+09 sec, polling station 0, elderly/pregnant: 18
At 1.68469e+09 sec, polling station 1, elderly/pregnant: 15
At 1.68469e+09 sec, polling station 0, ordinary: 19
At 1.68469e+09 sec, polling station 1, elderly/pregnant: 16
At 1.68469e+09 sec, polling station 0, ordinary: 20
At 1.68469e+09 sec, polling station 2, elderly/pregnant: 8
At 1.68469e+09 sec, polling station 0, ordinary: 21
At 1.68469e+09 sec, polling station 1, elderly/pregnant: 17
At 1.68469e+09 sec, polling station 0, elderly/pregnant: 22
At 1.68469e+09 sec, polling station 2, elderly/pregnant: 9
At 1.68469e+09 sec, polling station 0, elderly/pregnant: 23
At 1.68469e+09 sec, polling station 1, elderly/pregnant: 18
At 1.68469e+09 sec, polling station 0, ordinary: 24
At 1.68469e+09 sec, polling station 1, elderly/pregnant: 19
At 1.68469e+09 sec, polling station 0, elderly/pregnant: 25
At 1.68469e+09 sec, polling station 1, ordinary: 20
At 1.68469e+09 sec, polling station 0, ordinary: 26
At 1.68469e+09 sec, polling station 1, ordinary: 21
At 1.68469e+09 sec, polling station 2, ordinary: 10
At 20 sec total votes: (Mary: 19, John: 9, Anna: 32)
Polling station 0 results:
Votes for Mary: 9
Votes for John: 4
Votes for Anna: 14
Polling station 1 results:
Votes for Mary: 6
Votes for John: 4
Votes for Anna: 12
Polling station 2 results:
Votes for Mary: 4
Votes for John: 1
Votes for Anna: 6
root@server:~/Desktop/project-2-los-shellos-hermanos#
```

Last Output For PART I - II - III -IV:

After implementing all parts correctly, we ran our program with;

./electronic_voting -t 60 -p 0.5 -f 0.1 -c 3 and get this output after it without any bugs or errors

On that SS, we can easily see different polling stations 0 - 1 - 2, based on our argument " -c ".

```
root@server:~/Desktop# cd project-2-los-shellos-hermanos/
root@server:~/Desktop/project-2-los-shellos-hermanos# ls
electronic_voting  main.cpp  README.md  sleep.cpp  sleep.hh
root@server:~/Desktop/project-2-los-shellos-hermanos# ./electron
-p 0.5 -f 0.1 -c 3
At 1.68467e+09 sec, polling station 0, ordinary: 0
At 1.68467e+09 sec, polling station 1, ordinary: 0
At 1.68467e+09 sec, polling station 0, ordinary: 1
At 1.68467e+09 sec, polling station 1, ordinary: 1
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 2
At 1.68467e+09 sec, polling station 1, ordinary: 2
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 3
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 3
At 1.68467e+09 sec, polling station 0, ordinary: 4
At 1.68467e+09 sec, polling station 1, ordinary: 4
At 1.68467e+09 sec, polling station 0, ordinary: 5
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 5
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 6
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 6
At 1.68467e+09 sec, polling station 0, ordinary: 7
At 1.68467e+09 sec, polling station 1, ordinary: 7
At 1.68467e+09 sec, polling station 0, ordinary: 8
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 8
At 1.68467e+09 sec, polling station 0, ordinary: 9
At 1.68467e+09 sec, polling station 1, ordinary: 9
At 20 sec total votes: (Mary: 6, John: 1, Anna: 13)
At 1.68467e+09 sec, polling station 0, ordinary: 10
At 1.68467e+09 sec, polling station 1, ordinary: 10
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 11
At 1.68467e+09 sec, polling station 1, ordinary: 11
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 12
At 1.68467e+09 sec, polling station 1, ordinary: 12
At 1.68467e+09 sec, polling station 0, ordinary: 13
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 13
At 1.68467e+09 sec, polling station 0, ordinary: 14
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 14
At 1.68467e+09 sec, polling station 0, ordinary: 15
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 15
At 1.68467e+09 sec, polling station 0, ordinary: 16
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 16
At 1.68467e+09 sec, polling station 0, ordinary: 17
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 17
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 18
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 18
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 19
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 19
At 20 sec total votes: (Mary: 16, John: 1, Anna: 23)
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 20
At 1.68467e+09 sec, polling station 1, ordinary: 20
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 21
```

```
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 21
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 22
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 22
At 1.68467e+09 sec, polling station 0, ordinary: 23
At 1.68467e+09 sec, polling station 1, ordinary: 23
At 1.68467e+09 sec, polling station 0, ordinary: 24
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 24
At 1.68467e+09 sec, polling station 0, ordinary: 25
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 25
At 1.68467e+09 sec, polling station 0, ordinary: 26
At 1.68467e+09 sec, polling station 1, ordinary: 26
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 27
At 1.68467e+09 sec, polling station 1, ordinary: 27
At 1.68467e+09 sec, polling station 0, ordinary: 28
At 1.68467e+09 sec, polling station 1, ordinary: 28
At 1.68467e+09 sec, polling station 0, elderly/pregnant: 29
At 1.68467e+09 sec, polling station 1, elderly/pregnant: 29
At 20 sec total votes: (Mary: 29, John: 4, Anna: 27)
Polling station 0 results:
Votes for Mary: 12
Votes for John: 2
Votes for Anna: 16
Polling station 1 results:
Votes for Mary: 17
Votes for John: 2
Votes for Anna: 11
Polling station 2 results:
Votes for Mary: 0
Votes for John: 0
Votes for Anna: 0
root@server:~/Desktop/project-2-los-shellos-hermanos#
```

Keeping Logs:

Explanation:

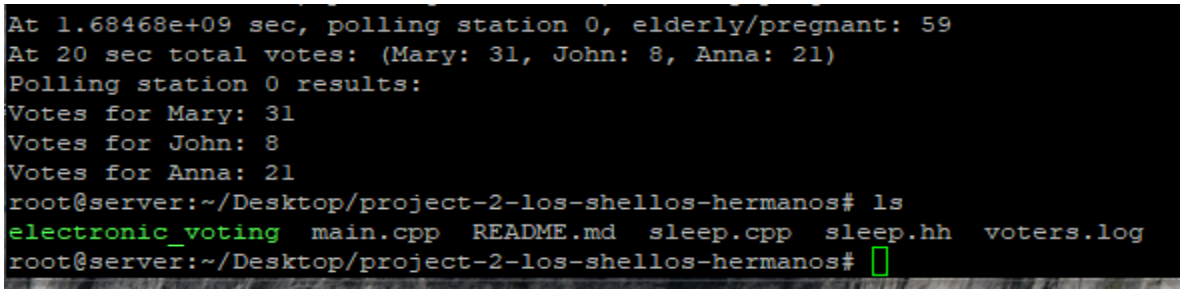
We declared request_time and polling_station_time variables with using time_t which is implemented in our *voter_process function. We recorded the current time when a voter requests their que position.

After that we created a VoterLogInfo log_info struct to keep relevant information.

Furthermore, we implemented a part of voter_log <<.... To write into our voters.log file.

Lastly, we implemented voter_log.open to our main function to create and open our voters.log file before voting and at the end of our main function we added voter_log.close to close our voters.log file.

After running our electronic voting system, we are able to keep logs on voters.log file:



And here is the example of voters.log inside: