Distributed-Mutual-Exclusion

Description

For this assignment, we need to build a simulation where only one client is allowed to execute the critical section at any given time. This will help prevent two or more processes from accessing the same data simultaneously. Problems can occur when multiple processes concurrently modify shared resources, leading to issues such as race conditions, data inconsistencies, and unpredictable program behavior. The implementation of mutual exclusion through the simulation aims to ensure orderly and synchronized access to critical sections, minimizing the likelihood of conflicts and maintaining the integrity of shared data.

Algorithm

To do this there are various algorithms we can use, we chose to simulate the Token Ring algorithm.
The token ring algorithm makes sure that each client can only access the critical section, if the client has the token. After it has exited the critical section, the client then gives it to the next client in line. In that way every client gets to have their turn in the critical section.

The simulation

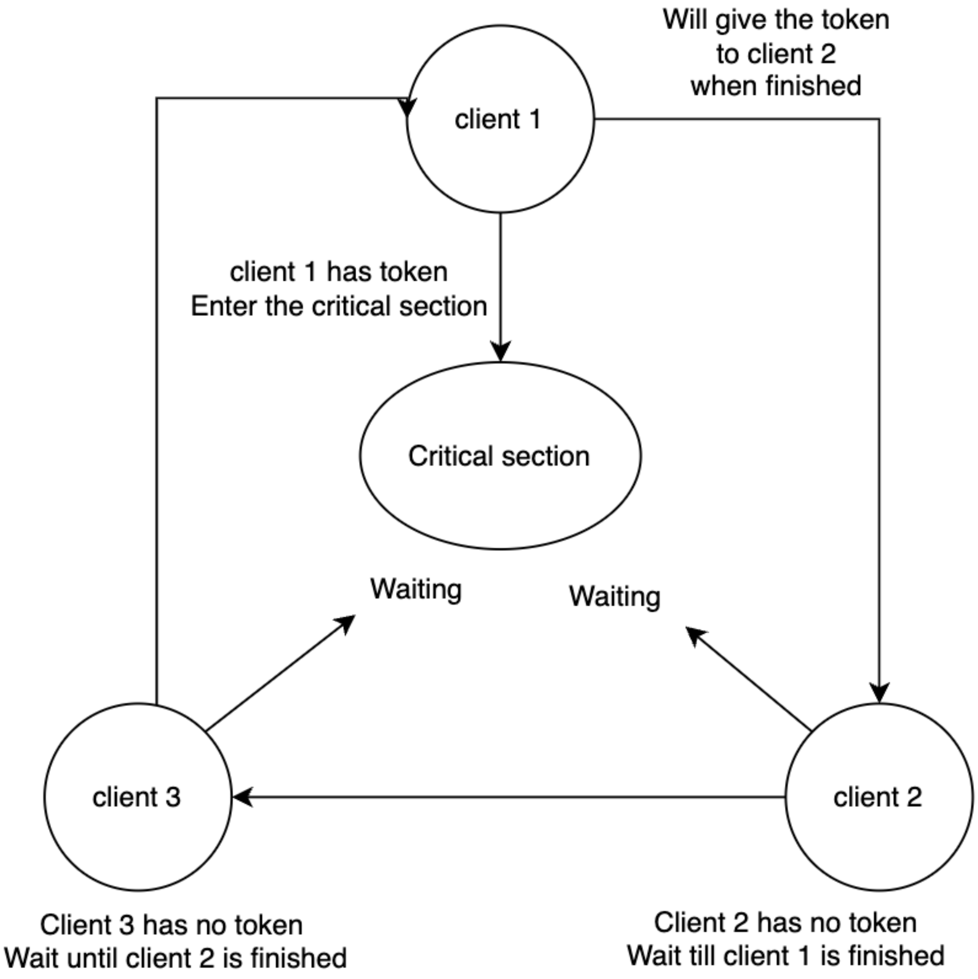
In our peer-to-peer system, each client functions as both a client and a server, eliminating the need for a central server. Once all clients are running and connected, messages can be exchanged, and clients can contend for entry into the critical section.

When a client seeks entry into the critical section, it checks if any other client is currently in the critical section. If not, it proceeds to enter the critical section by obtaining the token. The Token Ring algorithm incorporates a "nextClient" value. When a client finishes its critical section, it passes the token to its next client in the ring, allowing a sequential flow.

In our code, only one client is allowed to enter the critical section at a time, but the token is not passed to the next client in the circular sequence. In an ideal scenario, each client should possess a "nextClient" value, signifying the subsequent client in the queue. When multiple clients attempt to access the critical section, the process initiates with the first client gaining entry. After completion, this client hands over the token to the next client (client 2), who then follows the same process. This sequence continues until each client has entered the critical section without any conflicts.

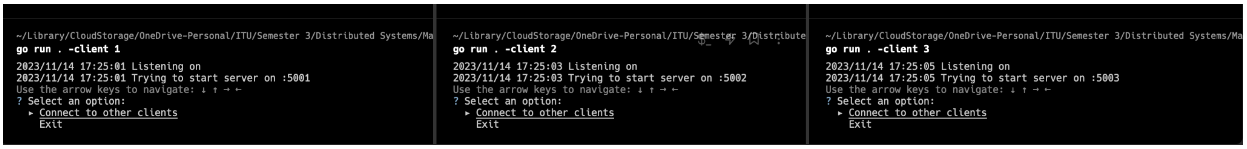
If, for instance, Client 2 possesses the token and enters the critical section, and Client 1 is waiting while Client 3 is idle, the token's natural flow in the Token Ring algorithm dictates that Client 2 releases the token upon completion. The token then circulates to the next client in the ring, which is Client 3. If Client 3 doesn't need access, it passes the token to the subsequent client, which, in this case, is Client 1. The order of token circulation is predefined and unaffected by the idleness of Client 3.

Here is an illustration:



Appendix

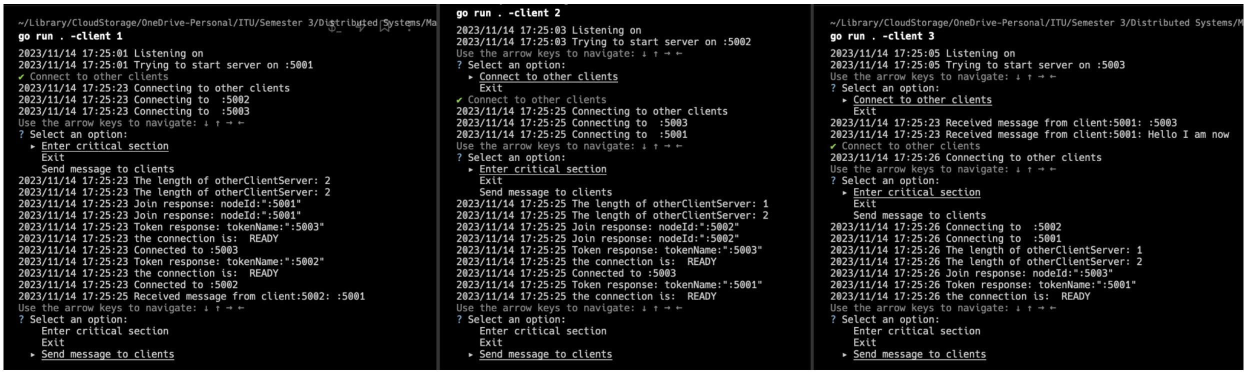
Link to the github repository: https://github.com/Josh0104/distributed-mutual



```
~/Library/CloudStorage/OneDrive-Personal/ITU/Semester 3/Distributed Systems/Ma
go run . -client 1
2023/11/14 17:25:01 Listening on
2023/11/14 17:25:01 Trying to start server on :5001
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
  ▸ Connect to other clients
    Exit
```

```
~/Library/CloudStorage/OneDrive-Personal/ITU/Semester 3/Distribute
go run . -client 2
2023/11/14 17:25:03 Listening on
2023/11/14 17:25:03 Trying to start server on :5002
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
  ▸ Connect to other clients
    Exit
```

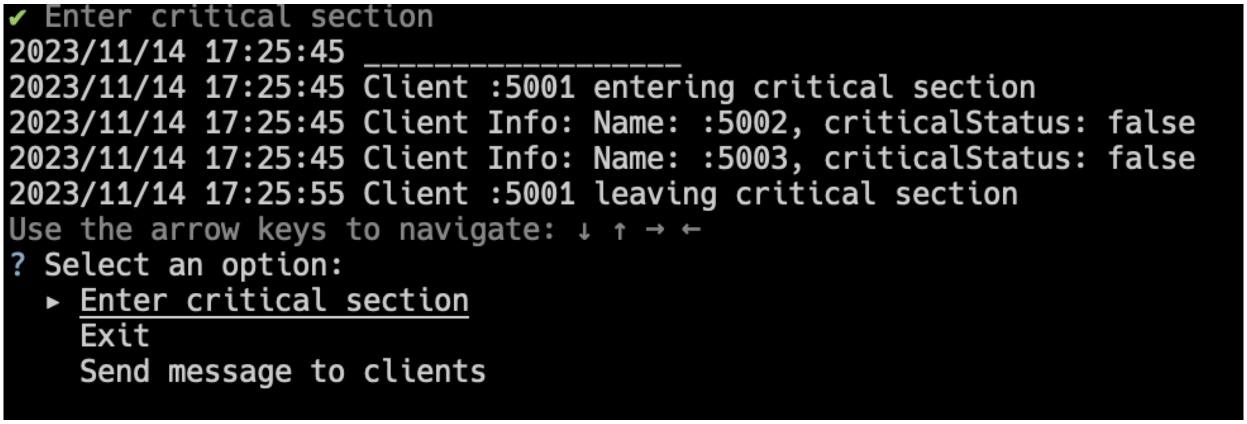
```
~/Library/CloudStorage/OneDrive-Personal/ITU/Semester 3/Distributed Systems/Ma
go run . -client 3
2023/11/14 17:25:05 Listening on
2023/11/14 17:25:05 Trying to start server on :5003
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
  ▸ Connect to other clients
    Exit
```



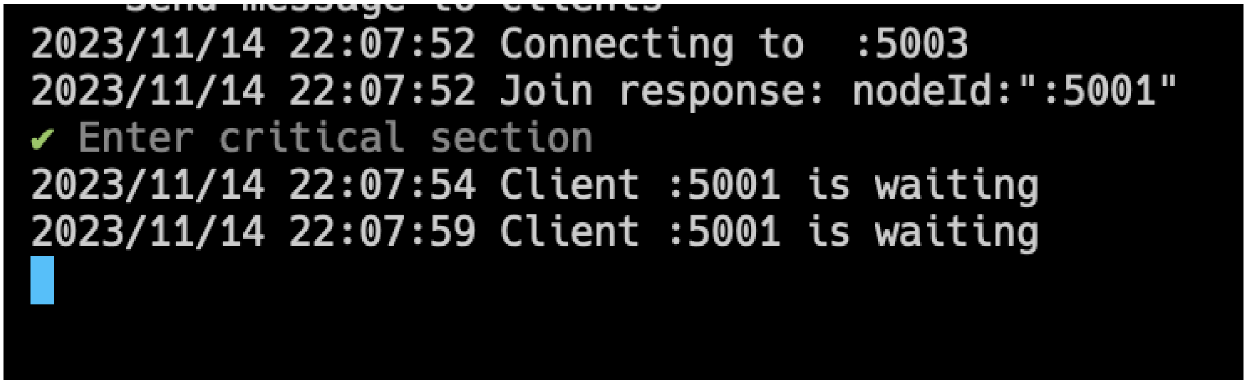
```
~/Library/CloudStorage/OneDrive-Personal/ITU/Semester 3/Distributed Systems/Ma
go run . -client 1
2023/11/14 17:25:01 Listening on
2023/11/14 17:25:01 Trying to start server on :5001
✔ Connect to other clients
2023/11/14 17:25:23 Connecting to other clients
2023/11/14 17:25:23 Connecting to  :5002
2023/11/14 17:25:23 Connecting to  :5003
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
  ▸ Enter critical section
    Exit
    Send message to clients
2023/11/14 17:25:23 The length of otherClientServer: 2
2023/11/14 17:25:23 The length of otherClientServer: 2
2023/11/14 17:25:23 Join response: nodeId:":5001"
2023/11/14 17:25:23 Join response: nodeId:":5001"
2023/11/14 17:25:23 Token response: tokenName:":5003"
2023/11/14 17:25:23 the connection is:  READY
2023/11/14 17:25:23 Connected to :5003
2023/11/14 17:25:23 Token response: tokenName:":5002"
2023/11/14 17:25:23 the connection is:  READY
2023/11/14 17:25:23 Connected to :5002
2023/11/14 17:25:25 Received message from client:5002: :5001
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
    Enter critical section
    Exit
  ▸ Send message to clients
```

```
go run . -client 2
2023/11/14 17:25:03 Listening on
2023/11/14 17:25:03 Trying to start server on :5002
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
  ▸ Connect to other clients
    Exit
✔ Connect to other clients
2023/11/14 17:25:25 Connecting to other clients
2023/11/14 17:25:25 Connecting to  :5003
2023/11/14 17:25:25 Connecting to  :5001
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
  ▸ Enter critical section
    Exit
    Send message to clients
2023/11/14 17:25:25 The length of otherClientServer: 1
2023/11/14 17:25:25 The length of otherClientServer: 2
2023/11/14 17:25:25 Join response: nodeId:":5002"
2023/11/14 17:25:25 Join response: nodeId:":5002"
2023/11/14 17:25:25 Token response: tokenName:":5003"
2023/11/14 17:25:25 the connection is:  READY
2023/11/14 17:25:25 Connected to :5003
2023/11/14 17:25:25 Token response: tokenName:":5001"
2023/11/14 17:25:25 the connection is:  READY
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
    Enter critical section
    Exit
  ▸ Send message to clients
```

```
~/Library/CloudStorage/OneDrive-Personal/ITU/Semester 3/Distributed Systems/M
go run . -client 3
2023/11/14 17:25:05 Listening on
2023/11/14 17:25:05 Trying to start server on :5003
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
  ▸ Connect to other clients
    Exit
2023/11/14 17:25:23 Received message from client:5001: :5003
2023/11/14 17:25:23 Received message from client:5001: Hello I am now
✔ Connect to other clients
2023/11/14 17:25:26 Connecting to other clients
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
  ▸ Enter critical section
    Exit
    Send message to clients
2023/11/14 17:25:26 Connecting to  :5002
2023/11/14 17:25:26 Connecting to  :5001
2023/11/14 17:25:26 The length of otherClientServer: 1
2023/11/14 17:25:26 The length of otherClientServer: 2
2023/11/14 17:25:26 Join response: nodeId:":5003"
2023/11/14 17:25:26 Token response: tokenName:":5001"
2023/11/14 17:25:26 the connection is:  READY
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
    Enter critical section
    Exit
  ▸ Send message to clients
```



```
✔ Enter critical section
2023/11/14 17:25:45 __________________
2023/11/14 17:25:45 Client :5001 entering critical section
2023/11/14 17:25:45 Client Info: Name: :5002, criticalStatus: false
2023/11/14 17:25:45 Client Info: Name: :5003, criticalStatus: false
2023/11/14 17:25:55 Client :5001 leaving critical section
Use the arrow keys to navigate: ↓ ↑ → ←
? Select an option:
  ▸ Enter critical section
    Exit
    Send message to clients
```



```
    Send message to clients
2023/11/14 22:07:52 Connecting to  :5003
2023/11/14 22:07:52 Join response: nodeId:":5001"
✔ Enter critical section
2023/11/14 22:07:54 Client :5001 is waiting
2023/11/14 22:07:59 Client :5001 is waiting
```