Erkan Dogan
2166262
2023-05-01

# HW1 – Socket Programming

## Abstract

In this homework, we aimed to develop a proxy server with cache functionality for clients to access and update the data table stored on a different server and programming language. The proxy server is responsible for caching frequently accessed data and handling client requests, such as GET, PUT, CLR, ADD, **DEL, SRH, FLS** operations. This report provides an overview of design and implementation and demo results.

## Design and Implementation

The homework consists of three main components: the client, the proxy server, and the main server. The client sends requests to the proxy server for data access and updates, the proxy server caches the data and handles the requests, and the main server stores the actual data table.

The proxy server was developed using Python, while the main server was developed using MATLAB. The cache on the proxy server was implemented as a queue with a fixed size, removing the oldest data when new data is added.

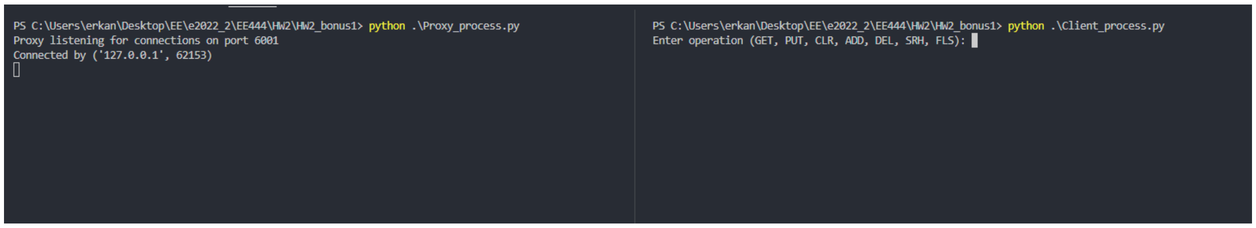
The following operations have been implemented for the scope of this homework. Bonus operations are indicated in bold. The "->" indicates that ENTER should be pressed for input parameters:

1. GET: Retrieve data from the table for specific indices (max 5 indices).
   a. To retrieve data, type GET followed by a comma-separated list of indices.
   (e.g., GET -> 1,2,3)
2. PUT: Insert or update data at the specified indices in the table (max 5 indices).
   a. To insert or update data, type PUT followed by a comma-separated list of indices, followed by comma-separated list of values.
   (e.g., PUT -> 5,2,7 -> 32,28,48)
3. CLR: Clear the entire data table, resetting it to its initial state.
   a. To clear the entire data table, simply type CLR and press Enter.
4. ADD: Add the data from specified indices and return the sum (max 5 indices).
   a. To sum the data at specified indices, type ADD followed by a comma-separated list of indices.
   (e.g., ADD -> 1,2,3)
5. **DEL: Delete the data from specified indices in the table (max 5 indices).**
   **a. To delete data at specified indices, type DEL followed by a comma-separated list of indices.**
   **(e.g., DEL -> 1,2,3)**
6. **SRH: Search for specific data in the cache or on the server and return the indices where the data is found.**
   **a. To search for specific data, type SRH followed by the data value you want to find.**
   **(e.g., SRH -> 5)**
7. **FLS: Flush/Clear the cache to remove all stored data.**
   **a. To clear the cache, simply type FLS and press Enter.**

Erkan Dogan
2166262
2023-05-01

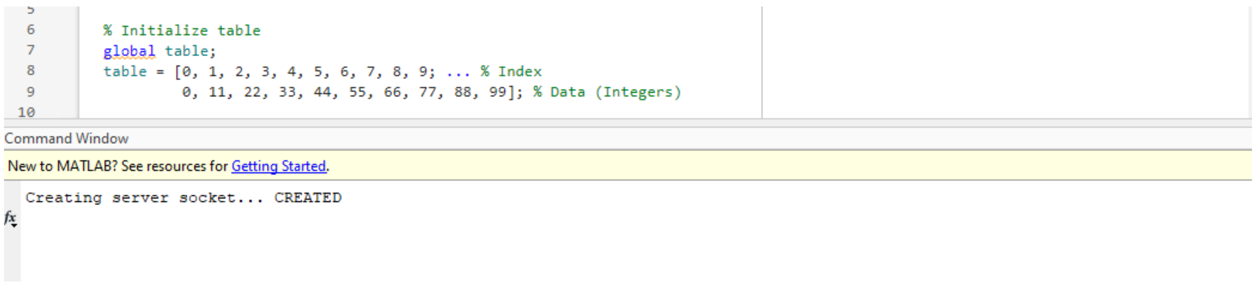
## DEMO

The implementation asks for a different number of inputs depending on the operation. Each of them has been explained in ‘Design and Implementation’ section. Due to the implementation limit, maximum level of indices is 5 for indices related inputs. The left side of the command window is Proxy_process.py and right side of the command window is Client_process.py.



```
PS C:\Users\erkan\Desktop\EE\e2022_2\EE444\HW2\HW2_bonus1> python .\Proxy_process.py
Proxy listening for connections on port 6001
Connected by ('127.0.0.1', 62153)
```

```
PS C:\Users\erkan\Desktop\EE\e2022_2\EE444\HW2\HW2_bonus1> python .\Client_process.py
Enter operation (GET, PUT, CLR, ADD, DEL, SRH, FLS):
```



```
% Initialize table
global table;
table = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9; ... % Index
         0, 11, 22, 33, 44, 55, 66, 77, 88, 99]; % Data (Integers)
```

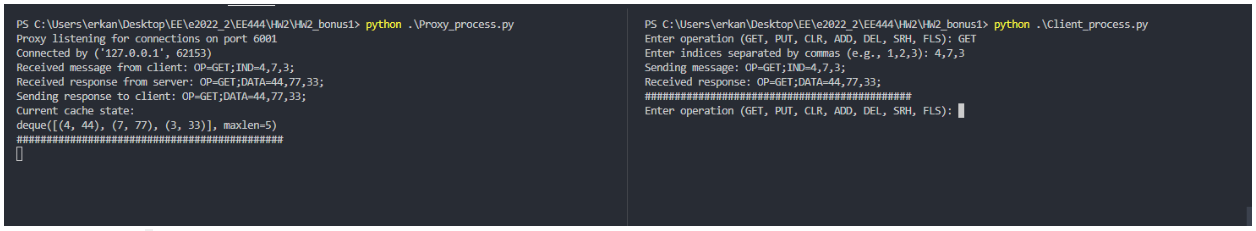
Command Window

New to MATLAB? See resources for Getting Started.

```
Creating server socket... CREATED
```

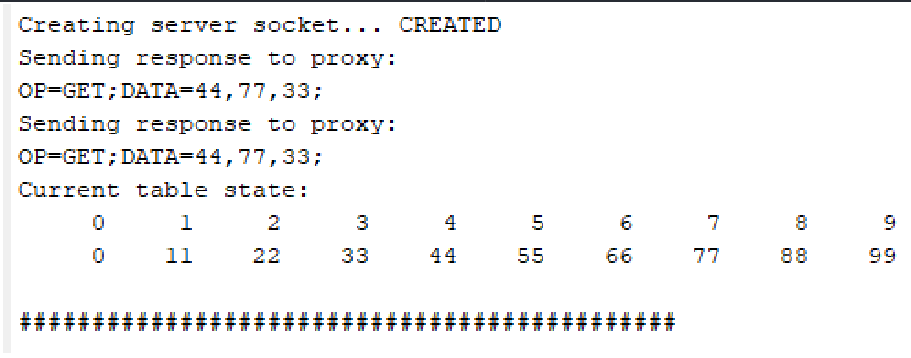
*Figure 1 The welcome screen and initial table data*

## 1. GET



```
PS C:\Users\erkan\Desktop\EE\e2022_2\EE444\HW2\HW2_bonus1> python .\Proxy_process.py
Proxy listening for connections on port 6001
Connected by ('127.0.0.1', 62153)
Received message from client: OP=GET;IND=4,7,3;
Received response from server: OP=GET;DATA=44,77,33;
Sending response to client: OP=GET;DATA=44,77,33;
Current cache state:
deque([(4, 44), (7, 77), (3, 33)], maxlen=5)
#############################################
```

```
PS C:\Users\erkan\Desktop\EE\e2022_2\EE444\HW2\HW2_bonus1> python .\Client_process.py
Enter operation (GET, PUT, CLR, ADD, DEL, SRH, FLS): GET
Enter indices separated by commas (e.g., 1,2,3): 4,7,3
Sending message: OP=GET;IND=4,7,3;
Received response: OP=GET;DATA=44,77,33;
#############################################
Enter operation (GET, PUT, CLR, ADD, DEL, SRH, FLS):
```

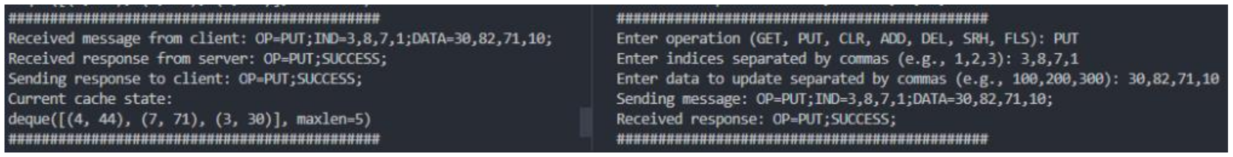


```
Creating server socket... CREATED
Sending response to proxy:
OP=GET;DATA=44,77,33;
Sending response to proxy:
OP=GET;DATA=44,77,33;
Current table state:
     0     1     2     3     4     5     6     7     8     9
     0    11    22    33    44    55    66    77    88    99

#############################################
```

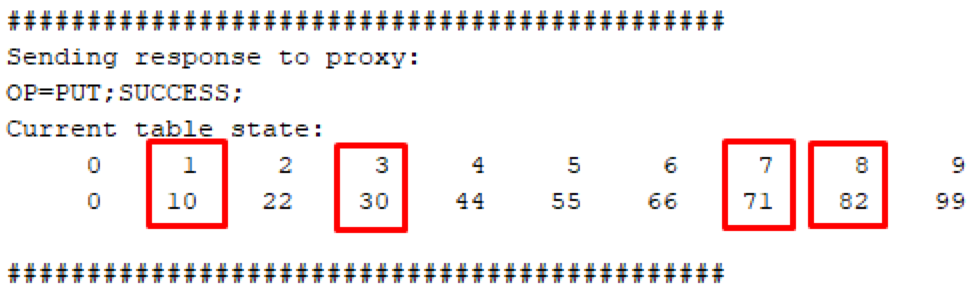
*Figure 2 GET operation*

## 2. PUT



```
############################################
Received message from client: OP=PUT;IND=3,8,7,1;DATA=30,82,71,10;
Received response from server: OP=PUT;SUCCESS;
Sending response to client: OP=PUT;SUCCESS;
Current cache state:
deque([(4, 44), (7, 71), (3, 30)], maxlen=5)
############################################
```

```
############################################
Enter operation (GET, PUT, CLR, ADD, DEL, SRH, FLS): PUT
Enter indices separated by commas (e.g., 1,2,3): 3,8,7,1
Enter data to update separated by commas (e.g., 100,200,300): 30,82,71,10
Sending message: OP=PUT;IND=3,8,7,1;DATA=30,82,71,10;
Received response: OP=PUT;SUCCESS;
############################################
```

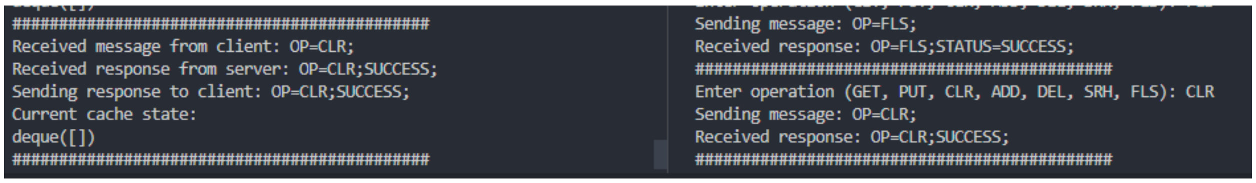


```
############################################
Sending response to proxy:
OP=PUT;SUCCESS;
Current table state:
     0     1     2     3     4     5     6     7     8     9
     0    10    22    30    44    55    66    71    82    99

############################################
```

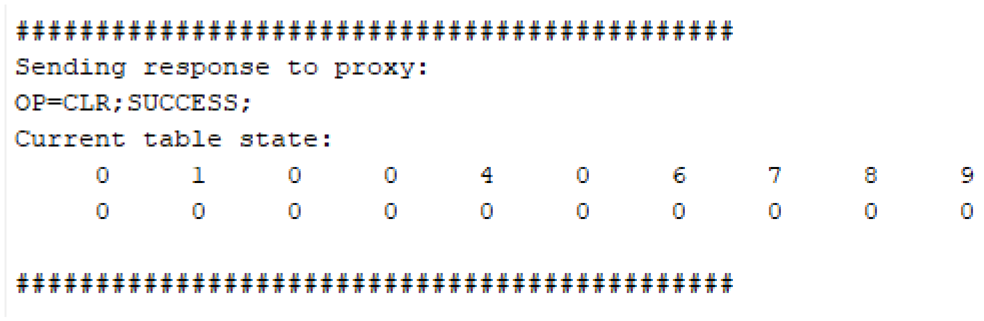
*Figure 3 PUT operation*

## 3. CLR



```
############################################
Received message from client: OP=CLR;
Received response from server: OP=CLR;SUCCESS;
Sending response to client: OP=CLR;SUCCESS;
Current cache state:
deque([])
############################################
```

```
Sending message: OP=FLS;
Received response: OP=FLS;STATUS=SUCCESS;
############################################
Enter operation (GET, PUT, CLR, ADD, DEL, SRH, FLS): CLR
Sending message: OP=CLR;
Received response: OP=CLR;SUCCESS;
############################################
```



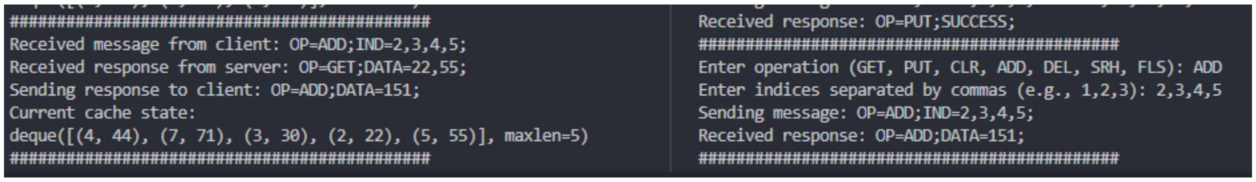
```
############################################
Sending response to proxy:
OP=CLR;SUCCESS;
Current table state:
     0     1     0     0     4     0     6     7     8     9
     0     0     0     0     0     0     0     0     0     0

############################################
```

*Figure 4 CLR operation*

Erkan Dogan
2166262
2023-05-01

## 4. ADD

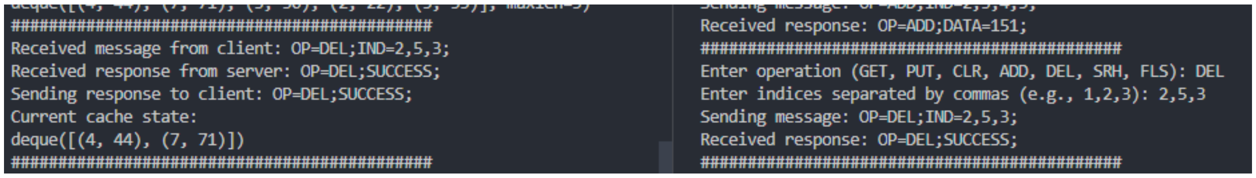


```
############################################
Received message from client: OP=ADD;IND=2,3,4,5;
Received response from server: OP=GET;DATA=22,55;
Sending response to client: OP=ADD;DATA=151;
Current cache state:
deque([(4, 44), (7, 71), (3, 30), (2, 22), (5, 55)], maxlen=5)
############################################
```

```
Received response: OP=PUT;SUCCESS;
############################################
Enter operation (GET, PUT, CLR, ADD, DEL, SRH, FLS): ADD
Enter indices separated by commas (e.g., 1,2,3): 2,3,4,5
Sending message: OP=ADD;IND=2,3,4,5;
Received response: OP=ADD;DATA=151;
############################################
```

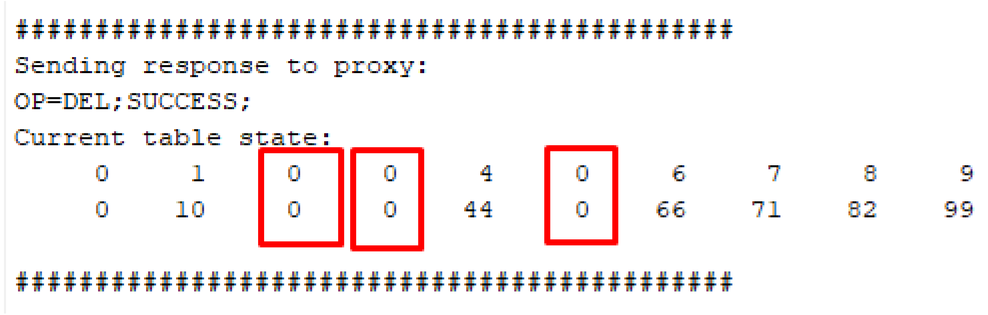
*Figure 5 ADD operation*

## 5. DEL



```
############################################
Received message from client: OP=DEL;IND=2,5,3;
Received response from server: OP=DEL;SUCCESS;
Sending response to client: OP=DEL;SUCCESS;
Current cache state:
deque([(4, 44), (7, 71)])
############################################
```

```
Received response: OP=ADD;DATA=151;
############################################
Enter operation (GET, PUT, CLR, ADD, DEL, SRH, FLS): DEL
Enter indices separated by commas (e.g., 1,2,3): 2,5,3
Sending message: OP=DEL;IND=2,5,3;
Received response: OP=DEL;SUCCESS;
############################################
```

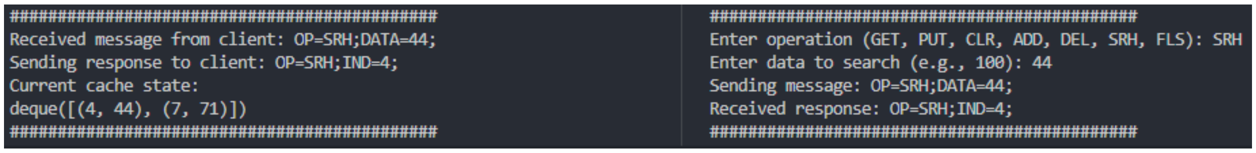


```
#############################################
Sending response to proxy:
OP=DEL;SUCCESS;
Current table state:
     0     1     0     0     4     0     6     7     8     9
     0    10     0     0    44     0    66    71    82    99

#############################################
```

*Figure 6 DEL operation*

## 6. SRH

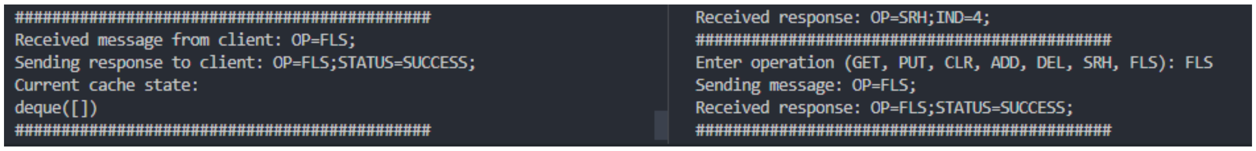


```
############################################
Received message from client: OP=SRH;DATA=44;
Sending response to client: OP=SRH;IND=4;
Current cache state:
deque([(4, 44), (7, 71)])
############################################
```

```
############################################
Enter operation (GET, PUT, CLR, ADD, DEL, SRH, FLS): SRH
Enter data to search (e.g., 100): 44
Sending message: OP=SRH;DATA=44;
Received response: OP=SRH;IND=4;
############################################
```

*Figure 7 SRH operation*

## 7. FLS



```
############################################
Received message from client: OP=FLS;
Sending response to client: OP=FLS;STATUS=SUCCESS;
Current cache state:
deque([])
############################################
```

```
Received response: OP=SRH;IND=4;
############################################
Enter operation (GET, PUT, CLR, ADD, DEL, SRH, FLS): FLS
Sending message: OP=FLS;
Received response: OP=FLS;STATUS=SUCCESS;
############################################
```

*Figure 8 FLS operation*