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).
   1. 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).
   1. 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.
   1. 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).
   1. 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).**
   1. **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.**
   1. **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.**
   1. **To clear the cache, simply type FLS and press Enter.**

# 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.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Figure The welcome screen and initial table data

# GET

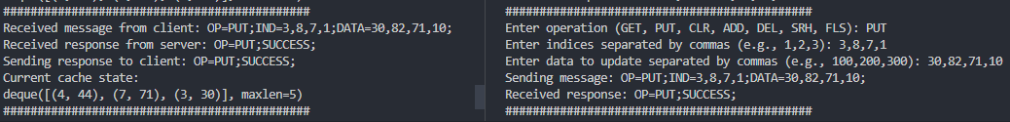
Text

Description automatically generatedChart, scatter chart

Description automatically generated

Figure GET operation

# PUT



A picture containing chart

Description automatically generated

Figure PUT operation

# CLR

Graphical user interface, application, Teams

Description automatically generated

Chart, scatter chart, box and whisker chart

Description automatically generated

Figure CLR operation

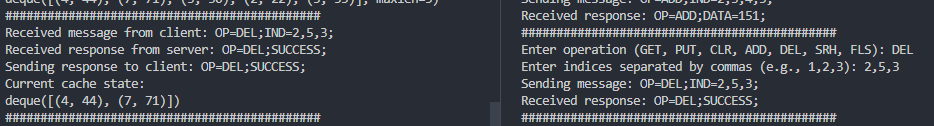
# ADD

Graphical user interface, text, application

Description automatically generated

Figure ADD operation

# DEL



Graphical user interface

Description automatically generated with medium confidence

Figure DEL operation

# SRH

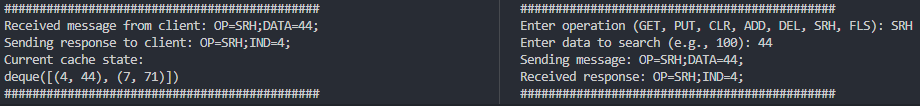


Figure SRH operation

# FLS



Figure FLS operation