TCP SERVER

Overview:
TCP server with a client handler that listens on port number. The server is designed to handle various client requests, including join requests, termination requests, and calculation requests. The server records all requests and responses in a log file named log.txt.

Server Functionality:
The server is built using Java and has a main method and a client handler class. The server socket is created using a ServerSocket object, which listens on a port for incoming connections. Whenever a new client connects, the server creates a new thread to handle that client's connection.

Client Handler Functionality:
The client handler class implements the Runnable interface and is responsible for handling multiple requests per client. It receives messages from the client, parses them, and determines the type of request. Based on the request type, it executes the appropriate handler method, which sends a response back to the client and logs the request and response in the log file.

Handler Methods:
There are three handler methods in the client handler class: handleJoinRequest, handleTerminationRequest, and handleCalculationRequest.

handleJoinRequest: executed when a client sends a join request to the server. It sends a join acknowledgment back to the client, starts a timer to record the client's connection duration, and logs the request and response in the log file.

handleTerminationRequest: executed when a client sends a termination request to the server. It sends a termination acknowledgment back to the client, stops the timer, calculates the client's connection duration, and logs the request and response in the log file.

handleCalculationRequest: executed when a client sends a calculation request to the server. It performs the basic mathematical operation, sends the result back to the client, calculates the client's connection duration, and logs the request and response in the log file.

Log File:
All client requests and responses are logged in a file named log.txt. The log file is created if it doesn't exist and is appended if it does. Each log entry includes the client's name, the current date and time, the request or response message, and the client's connection duration. (for JoinRequest client’s connection duration is excluded)

TCP CLIENT

Overview:
The TCP client initiates a connection with the server and makes as many calculation requests as it wants (minimum of 3) and terminates the connection when it doesn't wish to make any more requests.

Client Functionality:
The client is built using Java and consists of a main method and helper functions. The client socket is created using a Socket object, which is then used to communicate with the server.

The client is first given an option to connect to the server. If they choose to connect, then they are asked to input a client name to identify them, then a join request is sent to the server and upon acknowledgment of the request the client can start making calculation requests. If they choose not to connect, then the client socket is closed and the client is terminated.

The client is required to make at least 3 calculation requests, so after successfully connecting to the server, the client is asked to submit those requests one after another, and after each request it receives an answer from the server. The client is then able to make more calculation requests if it wants to.

Once the client doesn't wish to make any more calculation requests, the client makes a termination request and upon receiving an acknowledgment of the request from the server, the connection is terminated, the client socket is closed, and the client is terminated.

Helper functions:
There are two such functions and these are used to ensure that the client input is valid and in the expected format.

validInput: every time the client is asked whether they wish to do something (connect to server or make a calculation request), the expected answer is either 'y' or 'n', capitalized or lowercase. The function returns true if that is the case, false otherwise.

validRequest: makes sure that the calculation request inputted by the client is in the correct format and is valid. Also ensures that no illegal math is attempted (division by zero).