Network Programming Assignment

Xu Ziyin 3036173372

For both `GameClient.py` and `GameServer.py`, there is a class `Stage`, listed out all possible stages for the player.

Server

First start `GameServer.py`, it will start a server. Then waiting for client connection. If the server receives a connection from specific port, it will start a thread calling `HandleUser()`, which works to parse and handle request from client and changes the stage of one player. Meanwhile, the main thread creates a new socket listening the server port to wait for next connection.

Functions used to handle users' request are listed below:

For `handleWaiting()`:

It is used to handle Waiting stage. In this stage, a client is waiting for another player entering the same room. So it will frequently acquire the lock, check user number of the room and release the lock. When the room is full, this function will notice the waiting client and start the game.

For `handleLogIn()`:

Check whether the Authentication information is correct. If so, return stage `InTheGameHall` and inform the client, else return Authorization stage.

For `handleList()`:

Used to handle request `"/list"`, which is sent by client to get the room number and the user number. It will acquire the lock, store and send the answer and then release the lock

For `handleEnter()`:

Used to handle event "User Enter Room" with room number. Firstly it will acquire the lock and the check if the room is full. The first client entering the room will change to Waiting stage and receive waiting response. The second client will start the game and inform its partner that the game starts. The third and later joined client can not join the room so stay InTheHall stage. The lock will be released finally.

For `handleGaming()`:

After the game started, both two clients and the threads in server are in Gaming stage. Every thread receives a answer of its client and then write it in the `RoomAnswer` list with the lock acquired. The first thread that writes the answer should release the lock and frequently acquires and releases the lock to check if its partner has written another answer. If two clients finish writing answer, the later one will call `gameOver` to judge the game's answer, respond two clients and clear room message when the earlier thread returns when the room is cleared

For `handleExit()`:

When the UserHandler receives a proper exit request, the thread will send the client exit response and then terminate itself.

Client

When starting the `GameClient.py`, it will start the client. While the client is not closed, it calls `handleClient()`.

If the current stage is Waiting, then `handleWait()` is called. Because at that time the client cannot send any messages until it receives a message. If the client is not waiting, call `respHandler()`, which returns the current stage according to the `resp[0]`.