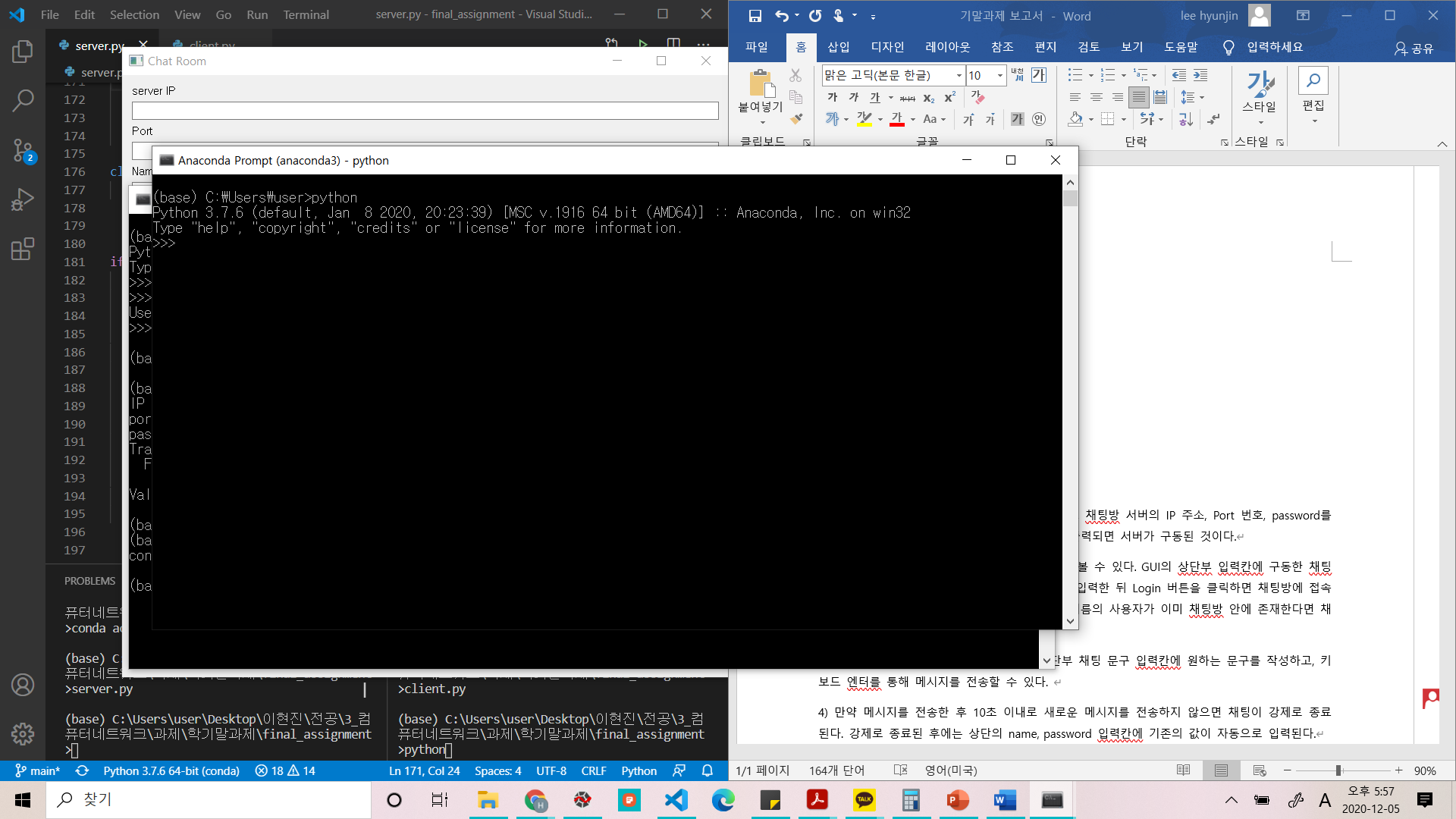
**조원: 2018147 이한별, 2018147535 이현진**

**1. 개발 환경**

- Python 3.7.6



- conda 4.9.2

- visual studio code 편집기 사용

**2. 실행 방법**

1) anaconda prompt에서 server.py를 실행시키고, 채팅방 서버의 IP 주소, Port 번호, password를 입력한다. prompt 창에 ‘run server’라는 문구가 출력되면 서버가 구동된 것이다.

2) client.py를 실행시키면 GUI 화면이 뜨는 것을 볼 수 있다. GUI의 상단부 입력칸에 구동한 채팅방 서버의 IP 주소, Port 번호, name, password를 입력한 뒤 Login 버튼을 클릭하면 채팅방에 접속할 수 있다. 만약 password가 다르거나 동일한 이름의 사용자가 이미 채팅방 안에 존재한다면 채팅방에 접속할 수 없다.

3) 채팅방에 정상적으로 연결이 되었다면 하단부 채팅 문구 입력칸에 원하는 문구를 작성하고, 키보드 엔터를 통해 메시지를 전송할 수 있다.

4) 만약 메시지를 전송한 후 10초 이내로 새로운 메시지를 전송하지 않으면 채팅이 강제로 종료된다. 강제로 종료된 후에는 상단의 name, password 입력칸에 기존의 값이 자동으로 입력된다.

5) 하단의 Quit 버튼을 클릭하면 채팅을 종료할 수 있다. 이미 채팅이 종료되거나 시작하지 않은 상태라면 Quit 버튼을 눌러도 아무 일도 일어나지 않는다.

6) 파일 전송방법 설명….

**3. 코드 설명**

1) server.py

socketserver 모듈과 TCPServer 클래스를 사용하여 서버를 구현하였다.

① main 함수

IP address, Port number, password를 입력 받고, 해당 IP, Port 값을 가지는 서버를 생성한 뒤, server\_forever() 함수를 통해 서버를 실행한다. 이 함수는 클라이언트의 접속 요청을 수신 대기하며, 접속 요청이 있을 경우 SocketHandle의 handle() 메소드를 호출한다. crtl+c가 입력되었을 경우 서버 구동을 종료한다.

② SocketHandle 클래스

- handle(self)

클라이언트의 접속요청이 있을 경우 실행하는 함수이다.

registerUsername() 함수를 실행하여 반환값을 username에 저장한다.

username이 ‘quit’라면 사용자가 ‘Quit’버튼을 클릭한 것이므로 채팅이 종료되었다는 메시지를 해당 사용자에게 전송하고, 사용자와의 연결을 끊는다.

username이 ‘false’라면 이미 등록된 사용자 이름을 입력했거나 잘못된 비밀번호를 입력한 것이므로 사용자와의 연결을 끊는다.

정확한 비밀번호와 유효한 이름이 입력되었다면 사용자와의 연결을 유지하고, while문을 통해 사용자로부터 온 메시지를 지속적으로 수신한다. 수신받은 메시지는 msg에 저장한 뒤, messageHandler() 함수의 매개변수 중 하나로 전달한다.

이 함수의 반환값이 -1이라면 사용자가 ‘Quit’ 버튼을 클릭한 것이므로 채팅이 종료되었다는 메시지를 해당 사용자에게 전송하고, 사용자와의 연결을 끊는다. 이 함수의 반환값이 -2이라면 시간이 초과되어 사용자가 강제 종료된 것이므로 시간이 초과되었다는 메시지를 해당 사용자에게 전송하고, 사용자와의 연결을 끊는다. 두 경우에 대해 break문을 사용하여 while문을 종료한다.

- registerUsername(self)

사용자의 이름을 등록하는 함수이다.

클라이언트로부터 이름과 비밀번호가 ‘이름’+’ ‘+’비밀번호’ 형태로 수신되기 때문에, 이름과 비밀번호를 분리한 뒤, addUser() 함수에 해당 이름과 비밀번호를 매개변수로 전달한다. 만약 isFalse나 isSame이 true라면 비밀번호가 틀렸거나 동일한 이름의 사용자가 채팅방에 이미 존재하는 것이므로 ‘false’를 반환한다.

- getfilefromclient(self, fn)

- sendfiletoclient(self, fn, un)

③ userHandle 클래스

- users

채팅방 사용자 정보가 담길 dictionary이다. {username: (socket, address), …} 형태이다.

- addUser(self, username, password, conn, addr)

사용자를 users에 추가하는 함수이다.

매개변수로 받은 username이 users 안에 있는지 확인하여, 이미 등록된 사용자인지 확인한다. 이미 등록된 사용자라면, TimeOverUsers 배열에서 강제로 종료된 사용자인지 확인하여 users에 저장되어 있는 해당 username의 socket과 address를 수정한다. 그 후 재접속 메시지를 채팅방 모두에게 전송한다. 강제로 종료되지 않은 사용자라면 이미 등록되었다는 메시지를 해당 사용자에게 전송한다. 이미 등록되지 않은 사용자라면 해당 username의 socket과 address를 users에 추가하고 접속 메시지를 채팅방 모두에게 전송한다.

thread를 사용하여 여러 명의 사용자들을 관리하기 때문에, thread 동기화를 막기 위해 lock을 사용하여 users를 수정하는 도중에는 해당 thread를 제외한 다른 thread는 접근하지 못하도록 한다.

- deleteUser(self, username)

사용자를 users에 삭제하는 함수이다.

username이 users안에 있다면 해당 원소를 users에서 삭제한다.

thread를 사용하여 여러 명의 사용자들을 관리하기 때문에, thread 동기화를 막기 위해 lock을 사용하여 users를 수정하는 도중에는 해당 thread를 제외한 다른 thread는 접근하지 못하도록 한다.

- messageHandler(self, username, msg)

사용자로부터 받은 메시지를 처리하는 함수이다.

사용자로부터 ‘quit’을 받았다면 해당 사용자를 users에서 삭제하고, -1을 반환한다.

사용자로부터 ‘timeover’을 받았다면 해당 사용자를 TimeOverUsers 배열에 넣은 뒤 -2를 반환한다.

- SendMsgAll(self, msg)

채팅방에 접속해있는 모든 사용자에게 메시지를 전송하는 함수이다. users 안에 들어있는 모든 사용자에 대해 send() 함수를 이용하여 msg를 전송한다.

2) client.py

PyQt5 모듈을 이용하여 채팅방 GUI를 구현하였다.

Thread 모듈을 이용하여 사용자마다 thread를 생성하고, 다중접속 채팅방 기능을 구현하였다.

time 모듈을 이용하여 사용자가 다음 메시지를 보내기까지 걸린 시간을 측정하였다.

TimeLimit 전역변수의 값을 변경하여 채팅방이 강제로 종료되는 기준 시간 간격을 설정할 수 있다.

① main 함수

ChatRoom 클래스를 이용하여 만든 인스턴스를 통해 채팅방 GUI를 실행한다.

② ChatRoom 클래스

- initUI(self)

GUI를 구성하는 함수이다.

QLable, QLineEdit, QPushButton, QTextBrowser, QVBoxLayout을 이용하여 IP 주소, Port 번호, 이름, password, 채팅 메세지 입력칸과, Login·Quit 버튼, 그리고 채팅방 화면을 구현하였다.

- onLoginClick(self)

사용자가 Login 버튼을 클릭했을 때 호출되는 함수이다.

IP 주소, Port 번호, 이름, password 입력칸에 입력된 값을 가져오고, 입력칸은 초기화한다. 가져온 IP 주소와 Port 번호를 이용하여 소켓 연결을 한다. 그리고 runChat() 함수를 호출한 뒤 이름과 password 값을 ‘이름’+’ ’+’비밀번호’ 형태로 만들어서 서버에 전송한다. isLogin을 true로 변경하여 채팅방에 접속되었다는 표시를 한다. 그 후 time() 함수를 이용하여 시간 측정을 시작한다.

- onQuit(self)

사용자가 Quit 버튼을 클릭했을 때 호출되는 함수이다.

채팅방에 접속되어 있는 상태라면 서버로 ‘quit’ 메시지를 전송한다. 그 후 isLogin을 false로 변경하여 채팅방이 종료되었다는 표시를 한다. 접속되어 있지 않은 상태라면 아무 일도 일어나지 않는다.

- getText(self)

사용자가 채팅 메시지 입력칸에 메시지를 입력한 뒤 엔터를 눌렀을 때 호출되는 함수이다.

time() 함수를 사용하여 시간 측정을 시작한 이후로부터 얼마만큼의 시간이 소요되었는지 측정하여 timeTerm에 저장한다.

만약 timeTerm이 TimeLimit보다 크다면, 기준 시간 간격보다 초과된 것이므로 사용자는 채팅방에서 강제 종료되어야 한다. isLogin이 true라면 현재 채팅방에 접속되어 있는 상태이므로 서버에 ‘timeover’ 메시지를 전송하고, 이름과 password 입력칸에 해당 사용자의 이름, password값을 설정한다. 그 후 isLogin을 false로 변경하여 채팅방이 종료되었다는 표시를 한다. 만약 isLogin이 false라면 채팅방에 접속하지 않은 상태이기 때문에 아무런 일도 일어나지 않는다.

timeTerm이 TimeLimit보다 작다면, 기준 시간 간격을 초과하지 않았으므로 정상적으로 메시지를 서버에 전송할 수 있다. 채팅 메시지 입력칸에 입력된 메시지를 가져와서 서버에 전송한다.

마지막으로 채팅 메시지 입력칸을 초기화한다.

- sendFileToServer(self, msg)

- getFileFromServer(self, filename)

③ rcvMsg(sock, ex)

④ runChat(ex, sock)

thread에서 rcvMsg 함수를 실행한다. 매개변수로 sock과 ex를 전달한다.

**4. 참고 사이트**

[1] socketserver, <https://brownbears.tistory.com/207>

[2] socketserver, <https://ncookie.tistory.com/46>

[3] 소켓 종료 명령어,

<https://eastroot1590.tistory.com/entry/%EC%86%8C%EC%BC%93-%EC%A2%85%EB%A3%8C-%EB%AA%85%EB%A0%B9-close%EC%99%80-shutdown>

[4] thread를 이용한 채팅방 구현, <https://webnautes.tistory.com/1381>

[5]