Python Homework Day 09 - Sockets

Guess Server:

1) Create a Guess-the-Number server. Using UDP. The server will pick a random number from 1 to 1000. clients will be able to send a number to the server and will get a response: "too big" / "too small" / "correct". if correct, the server will choose another random number.

2) make a version of the server that sends the number to guess as binary (your choice of byte size and endianness)

3) create a TCP version of Guess-the-number server where each client connects and gets its own random number to guess and will also have a number of guesses before failing. you will need to find a way to store the random number along with its client data, for example you could use a tuple, or make a class to hold the session data (socket, address, number_to_guess, guesses_remaining)

Rock Paper Scissors:

4) Create a program that can act as both a server and a client (ask the user if they wish to host a game or connect to a game). as a server the program will wait for a connection and will then play a game with the client. as a client the program will connect to a server and then play a game with the server. on both the user will be asked to input rock/paper/scissors (r/p/s) after both have made a choice the result will be printed to both, the winner side will print ("you win") the loser side will print ("you lose"), on a tie print:"tie". repeat 3 times and print the winner/loser/tie of best of 3 and disconnect both.

5) create a server only code that will act as a pairing server, create a client version that connects to that server. when two clients connect the server will pair them, making one act as host and the other as client to play against each other.

Login Server:

6) Create a server program with a dictionary of emails as keys and passwords as values, fill it with some values. create a client program that will allow registering or logging in.
    a) For registration, the user will be asked to enter an email, which will be sent to the server. if it is already taken, tell print a message to the client and return to the "register/login" menu. if the email is valid the user will enter a password twice and the password will be sent to the server.

    b) For Logging in, the user will enter email and password, on success print "successfully logged in", on failure print "bad credentials"