COP4521 Programming Assignment 4: Internet Chat Server

Objectives:

- Practice socket programming in Python
- Practice network application development

Description:

This assignment can be done by a group of two students (or individually if you prefer). You are strongly encouraged to form a group.

In this assignment, you will implement a basic Internet chat room server that allows users to communicate with one another individually or in groups. The server supports the basic functionality of a typical Internet chat room server:

1. Users can log in as guests and register a username and password for the server.
2. After a user registers, they can log in with their username and password.
3. Once a user logs in to the server, they may start a new discussion topic by creating a room. Other users may join or leave the room and participate in the group discussion.
4. A user can:
   a) Communicate one-on-one with another user,
   b) Join the discussion in a room,
   c) Broadcast messages to all currently online users.
5. The server should provide some utility functions, such as listing all online users and displaying information about rooms and users.

More specifically, the following commands should be supported:

[] optional field, <> required field

```
who                          # List all online users
status [user]                # Display user information
start <topic>                # Start a room for a topic
rooms                        # List all current rooms
join <room number>           # Join a room
leave <room number>          # Leave a room
shout <message>              # Broadcast <message> to everyone online
tell <user> <message>        # Tell <message> only to the user
info [Info_text]             # Change or show your information text
quit                         # Logout
exit                         # Logout
block  <user>                # block a user
unblock <user>               # unblock a user
say <room number> <msg>      # Broadcast <msg> to everyone in room <room number>
help                         # Print this message
register <user> <passwd>     # Register a new user
```

- The server should work with a plain Telnet client.
- Anyone can log in using a guest account to register.
- Both guest and regular accounts can register new users.
- Account information should be retained across sessions, even if the server crashes.
- The user who starts a topic becomes the leader of the room. When the room leader leaves the room or exits the system, the room should be closed.
- A user can be in multiple rooms at the same time.
- Only users who are in a room can speak (using the `say` command) and listen to the discussion in that room.
- The server should not crash due to user behavior. No matter what a particular user does, the server must continue to serve other users.
- A user will not receive any messages (via `tell`, `shout`, or `say`) from a blocked user.
- If User A is blocked by User B, the block will persist across sessions until User B unblocks User A.
- The server should provide basic error feedback to users, including the following:
    - Login error (username/password do not match)
    - Unsupported command
    - Incorrect command format
    - User does not exist (for commands that specify a user)
    - User is not online
    - Room does not exist
    - Attempting to speak in a room without being a member of that room

A sample executable is running on linprog6 port 55555, 55556, 55557, 55558, 55559. 'telnet linprog6 55555' from a department server to use the sample chat room. You should try to make your server behave like the sample server (while fixing bugs in the sample server).

Due time: October 31, 2025, 11:59pm.

Submission: Name your program using the format: **lastname_firstinitial_assignment4.py.** Put all of your program and necessary files in a tar file called **lastname_firstinitial_assignment4.tar** and submit the tar file on Canvas.

Grading (70 points total):

- Include the basic header (template available on the course website) for the assignment. Name your program lastname_firstinitial_assignment4.py. Write the names of group members in the header. Also report your testing results for each grading items in the header in the three categories (pass, fail, partial). (10 points)
- register (5 pts)
- who (5 pts)
- status, info (5 pts)
- start, rooms, join, leave (10 pts)
- say (5 pts)

- shout (5 pts)
- tell (5 pts)
- block, unblock (5 pts)
- help, quit, exit (5 pts)
- error handling/feedbacks (5 pts)
- overall (5 pts)
- **Extra points:** The first person to report a bug in the sample program will receive **2 extra points**. Functional bugs only please. In cases that you crash the server, please give the deterministic sequence of commands to crash the server.

Notes:

- You should start with the sample server code provided (assignment4_provided.py). The sample server contains about 400 lines of code.
- You should focus on the correct user behavior. You will notice many unspecified features in the project (e.g. a second connection from the same user). Your server needs to exhibit reasonable behavior in all cases. Under no circumstances should the server crash.
- Be mindful about race condition in the multithreading code.