The purpose of this exercise is to implement server-client communication using C#, expand MUD Multi-User Dungeon game features, and enhance understanding of network programming and object-oriented programming.

---------------------------------------------

Provided Code:

* Server: [Program.cs](https://kmuackr-my.sharepoint.com/:u:/g/personal/102113_ms_kmu_ac_kr/EcCG2U-R_LFItC46uuDSp4oB_miHvzwRNr2qUAp4a5gA4A?e=zqsm02) (Ctrl + Click)
* Client: [Program.cs](https://kmuackr-my.sharepoint.com/:u:/g/personal/102113_ms_kmu_ac_kr/EQONNXuYtyVHorjBniVDiKwBUjzicALBNm6UoXindTtddg?e=d30pyo) (Ctrl + Click)

---------------------------------------------

- Exercise Questions (Total: 18 points):

1. [**3pt**] There is no Main function on the client side. Create the Program class. Insert the Main function, and within Main, instantiate a MUDClient object and call StartGame(). (Completing the Program class on the client side will enable basic server-client communication.)
2. [**3pt**] The show command does not currently display the positions of players and items on the map. Modify the code so that the first player is marked as "1," the second player as "2," the Fart Bag as "F," and Medicine as "M" on the map.
3. [**3점**] When a player loses, modify the code so that only the losing player disconnects from the server.
   1. The winning player remains on the server, waiting for another player.
4. [**3pt**] Add "Trap" and "Health Fountain" items that can affect HP as players move through the room.
   1. "Trap" decreases HP by 10 when a player steps on it.
   2. "Health Fountain" restores HP by 15 (HP cannot exceed 100).
   3. Display new items as "T" and "H" in ShowRoomMap.
5. [**3pt**] Add a *say* command allowing the two players connected to the server to chat.
   1. Clients enter a text message after the *say* command. For example, entering *say Hello* would display "Tom: Hello" to the other player.
   2. The server relays each player's message to the other player.
      1. Use a method similar to BroadcastMessage to transmit a player’s message to the other player.
6. [**3pt**] Add a scan command that allows players to "detect" the other player within a certain range. When using scan, players can view the distance and position of the other player.
   1. Example for calculating the distance between two coordinates (Manhattan distance): (Math.Abs(player1.x - player2.x) + Math.Abs(player1.y - player2.y))
7. [**Honorary**] Expand the game to multiple rounds, adding a system to determine the winner after a set number of rounds.
   1. The game consists of 3 rounds, and the player who wins each round earns 1 point.
   2. After 3 rounds, the player with the highest score is declared the winner.
   3. Use the result command to display the current round and each player’s score.
8. [**Honorary**] Dynamically set the room size to a random dimension for each game.
9. [**Honorary**] Add a "timer" feature that automatically recovers or decreases a player’s health after a certain period.
   1. The timer runs every 30 seconds, reducing each player’s health by 5.
   2. Health changes are sent to each player whenever the timer executes.

-------------------------------------------------------------------------------------------------------------------

**Submission Guidelines:**

* During the class, complete up to question 6, have your work checked, then you may leave.
* Post-practical session submission deadline: November 23.
* Submission requirements: A single Word (.docx), Hangul (.hwpx), or PDF (.pdf) file containing:
  + Final source code (in text format, not images)
  + Captured images of the server and clients running

■