For the next project, you will write a 2 player network game in which each player can be located at a different Internet host. What game you implement is up to you, as long as it is “non-trivial” and requires several moves to arrive at a winner. Regardless of the game, the project must adhere to the following requirements.

The game system will consist of two programs — a game player client and a game server. There will be one instance of the game server running on some Internet host and several instances of the player client distributed arbitrarily throughout the network.

The game server is always running on a particular machine on the network and is responsible for both pairing players who want to play the game and keeping track of the current state of all games being played. When a player wants to play the game, he initiates the game player client, which sends a message to the server specifying that a new player wants to be paired with another player to play the game. When the server has received messages from two such players, it pairs them up and begins an instance of the game. If the server has only received one request, it should send an acknowledgement to the client and tell it to wait for a later message that will specify his partner and start the game.

When a player makes a game move, it is sent to the server. The server first verifies that the move is legal given the current state of the game. If the move is legal, the server updates its state and sends the new state to both players. Then the server waits for a move from the other player. If the move is not legal, the server tells the player this and waits for the player to make a legal move. If a legal move results in a player winning the game (or a tie), the server notifies both players and ends the game.

The server should be able to handle any number of games simultaneously. The server needs to be able to handle requests for new games at any time, while also moderating existing games. To do this, you will need to make use of the `select()` function.

You must use your reliable communication system from the third project for all communication. (You may modify the interface slightly if need be.) Notice that you will have to use the `control` field more generally to specify different kinds of messages (i.e., ACKs, new game requests, game moves, illegal move errors, state updates, etc.).

Your player program should accept one parameter on the command line — the host on which the server is running. The port on which the server is listening should be defined as a constant that is known to all players.

Your code should be nicely written and use object oriented design where applicable. Comments are also important!

In addition to your code, I will expect a detailed README document (just a plain text file is fine) that explains your design, the rationale for your design decisions, and how your system works. The purpose of this is twofold. First, it will help me to understand your code and your design. Second, it is meant to make you think carefully about your design, before you start coding.

This assignment may be completed with a partner. It is your responsibility to play nicely and divide up the work. Both partners will receive the same grade on the project, except in extreme circumstances. The grade will be based both on test cases and a read of your code. I will be looking for elegance and efficiency, in addition to correctness.

I will ask each partner to distribute 20 points between himself and his partner to indicate the relative amount of effort each devoted to the project. If you both worked on it equally hard, each should get 10 points. Each student should send me his or her points individually via email.