Exercise Set 7: Battleships

Exercise 7.1 The goal of this exercise is to use what you learned about data structures and class hierarchy design by implementing a battleship game to be played over the network.
The Battleship game is defined as follows.

Two players play against each other. Each has a board with 10 rows and 10 columns. At the beginning of the game, each player places the available battleships onto the board, hidden from the opponent's view. Players then take turns guessing the locations of the opponent's ships. The first player to destry all of the opponent's ships wins.

Design a program, which simulates one player playing against a human opponent, using the console for communications, as follows:

- When it is compute's turn, the computer selects a guess of one location on the board, and sends the message `guess(x, y)` to the opponent, where `x` and `y` are integers from `1` to the `size` of the board.
- The computer then awaits for the next message to arrive from the opponent.
- The opponent reply is through a message `reply(response-code)`, where the response code can be `HIT`, or `MISS`, or `WIN`, or `REPEAT` if the player had guessed the same location more than once.
  - If the response is `WIN`, the computer wins and the game stops.
  - If the response is `HIT`, the computer records the location of the hit in its board.
  - If the response is either `HIT` or `MISS`, the computer sends a message `yourTurn()` to the other player and awaits the opponent's move.
- The opponent (*for now a human player, but later will be another computer*proceeds in a similar fashion. When it receives the `guess(x, y)` message, it looks up the record of its ships, records the guess, and determines the answer.

The preliminary design of the record of the game and a way to play against a human with console input is available. The code needs *refactoring* to separate the communications part from the game record (see below). Try the code to play against a human player in the console.

- Define IOChannel interface, which allows you to read and write Strings - one line at a time. The interface contains methods open(), close(), String read(), and write(String).

- Refactor the battleship game, so all communications with the console is through ConsoleChannel which implements IOChannel interface.
- I will provide later today the code which modifies the number game to communicate through a similar channel. Use it as example to implement battleship game to work over a network socket connection.
- Design the methods needed for the computer to play the game. At the least, the computer may select a random location, which has not been guessed before.
- Design the methods needed to allow the user to select the location of the ships before the game starts.
- Make sure you have tests for each part of the game you have developed. (Do as I say...)