Game: One-Way Battleship

Description: Player needs to guess the location of five different battleships on a 10 X 10 board

These ships include include:

Carrier(5)

BattleShip(4)

Destroyer(3)

Submarine(3)

Patrol Boat(2)

The numbers indicate the amount of spaces the ship takes up.

Like the title says, this is one-way battleship. The user will win when they have destroyed all five battleships

The loser will lose when they run out of turns.

Turn limit is chosen by difficulty.

Easy: 80 turn limit

Medium 60 turn limit

Hard 40 turn limit

(Through testing multiple players it seems like these difficulties are in a good place)

Will be a pool of 3 preset game configurations.

Through serialization we can make it easier to store and load in boards, making our game configs much more diverse.

Could make randomized configs down the line, just want to focus on basic implementation now.

The program handles bad format when submitting targets through exception handling.

UML Class Diagrams (No hierarchy needed for this game)

Game (Main Model) (Implemented)

- gameBoard : Board

- gamePool : BoardPool

- status : string

- numTurns : int

- numHits : int

- numMiss: int

- turnLim : int

- user : string (Currently not being used, waiting for future project progression)

------

+ Initiallize(string difficulty, string user)

+ Status : string

+ NumTurns : int

+ NumMiss : int

+ NumHits: int

+ GameBoard : Board

+ DiffSettings(int easy, int med, int hard) (Not implemented yet; when users are implemented)

+ GuessLocation(string target)

+ VictoryStatus

Board (Data Structure and methods that deal with the gameBoard)

+ static SIZE : 10

+ static readonly rowNames : char[]

+ static readonly shipTypes : char[]

- charBoard : char[,]

- shipCheck : bool[]

------

+ ShipCheck : bool[]

+ GetSpot(string target) : char

+ SetSpot(string target, char newchar)

+ CheckBoard

+ CheckVictory

+ ToString

BoardPool (Pool of gameBoards)

- pool : Board[]

------

+ LoadBoards

+ LoadBoards(string fileName) (Not implemented yet)

+ GetRandomBoard : Board

View (Just main and console I/O)

Inputs:

Admin: (Admin controls not implemented yet)

New Boards (Board or char[,])

Difficulty Settings (string then an int)

Player:

Difficluty Choice (string)

Location(string of two-three chars)

Outputs:

Board Output(char[,] into a string; like a matrix)

'-' chars are "open waters"

'O' chars are confirmed empty spaces

'X' chars are hits

Hit or Miss confirmations (string)

Destroyed Ships confirmations (string)

End of Game confirmations (string)