ASSIGNMENT 10

REQUIREMENTS

- You will be given one of the problems below to solve.
- Use object oriented programming and layered architecture
- All modules with the exception of the UI will be covered with specification and **PyUnit** test cases.
- The program must protect itself against the user's invalid input.
- The deadline for this assignment is **week 13**.

BONUS POSSIBILITY (0.2P)

- In addition to the console-based user interface required, also implement a graphical user interface (GUI) for the program.
- To receive the bonus, both user interfaces (menu-based and graphical) must use the same program layers. You have to be able to start the application with either user interface.
- The deadline for the bonus is **week 13**.

PROBLEM STATEMENTS

1. CONNECT FOUR

The game is described here: https://en.wikipedia.org/wiki/Connect_Four

NB! We do not expect you to implement optimal play on the part of the computer player. However, it should still employ a strategy when making its moves in order to attempt to win the game and provide an entertaining opponent for the human player. Minimally, the computer player should move to win the game whenever possible and should block the human player's attempts at 1-move victory, whenever possible.

2. GOMOKU

The game is described here: https://en.wikipedia.org/wiki/Gomoku

NB! We do not expect you to implement optimal play on the part of the computer player. However, it should still employ a strategy when making its moves in order to attempt to win the game and provide an entertaining opponent for the human player. Minimally, the computer player should make its moves close to the area where stones have been placed, it should move to win the game whenever possible and should block the human player's attempts at 1-move victory, whenever possible.

3. OBSTRUCTION

The game is described here: http://www.papg.com/show?2XMX

Before the game starts, the user will provide the board size on which it will be played.

NB! We do not expect you to implement optimal play on the part of the computer player. However, it should still employ a strategy when making its moves in order to attempt to win the game and provide an entertaining opponent for the human player. Minimally, the computer player should move to win the game whenever possible.

4. BATTLESHIP

The game is described here: https://en.wikipedia.org/wiki/Battleship_(game)

The game will be played on an 8x8 board numbered A-H (columns) and 1-8 (lines). Each player will place 1 battleship (4 squares), 1 cruiser (3 squares) and 1 destroyer (2 squares). The computer places its ships randomly. The program will prevent both players from placing ships that reach out of the playing area or which overlap.

To help the player, the game will display both the player's grid as well as its "targeting" grid, that contains information about the player's 'hits' and 'misses'.

NB! We do not expect you to implement optimal play on the part of the computer player. However, it should still employ a strategy when making its moves in order to attempt to win the game and provide an entertaining opponent for the human player.

5. PLANES

The game is described here: https://ro.wikipedia.org/wiki/Avioane_(joc)

The game will be played on an 8x8 board numbered A-H (columns) and 1-8 (lines). Each player will place 2 planes having the shape as illustrated in the link above. The computer places its planes randomly. The program will prevent both players from placing planes that are outside the playing area or which overlap.

To help the player, the game will display both the player's grid as well as its "targeting" grid, that contains information about the player's 'hits' and 'misses'.

NB! We do not expect you to implement optimal play on the part of the computer player. However, it should still employ a strategy when making its moves in order to attempt to win the game and provide an entertaining opponent for the human player.