Exercise No. 4

Tricky five-in-a-row

Create a game, which is a variant of the well-known five-in-a-row game. The two players can play on a board consists of n x n fields. Players put their signs alternately (X and O) on the board. A sign can be put only onto a free field. The game ends, when the board is full, or a player won by having five adjacent signs in a row, column or diagonal. The program should show during the game who turns.

The trick in this variant is that if a player makes 3 adjacent signs (in a row, column or diagonal), then one of his signs is removed randomly (not necessary from this 3 signs). Similar happens, when the player makes 4 adjacent signs, but in this case two of his signs are removed.

Implement this game, and let the board size be selectable (6x6, 10x10, 14x14). The game should recognize if it is ended, and it has to show in a message box which player won (if the game is not ended with draw), and automatically begin a new game.

*Short description how to use the program:*

1. In BaseWindow class:

+. private void showExitConfirmation(): show Exit Cornfirmation.

+. protected void doUponExit(): causes the JFrame window to be destroyed and cleaned up by the operating system

1. In MainWindow class:

+. private ActionListener getActionListener(final int size): create Window for user

+. public List<Window> getWindowList(): return List of Window

1. In PlayerXO class:

+. public int getRow(): return index of Row

+. public int getColumn(): return index of Column

+. public void setRow(int row): set index of Row

+. public void setColumn(int column): set index of Column

1. In TicTacToe class: public static void main(String[] args): run the program
2. In Window class:

+. private void addButton(JPanel panel, final int i, final int j): add Button (in more detail, this function create the board of this game)

+. private void showGameOverMessage(Player winner): show game over message

+. private void showDrawMessage(): show draw message

+. private void newGame(): start new game

+. private void updateLabelText(): show which player is playing

1. In Model class:

+. public Player step(int row, int column): control the position of X and O

+. public Player findWinner(): find out X or O wins

+. public int findDraw(): decide if the game is draw or not

+. public boolean playTrick(): implement the condition of the task

+. public Player getActualPlayer(): return player X or O

*The connections between the events and event handlers:*

+. Implement functions

+. Add button to the Window (Board)

+. Create Window