1. description of the exercise

1. Connect Four

Connect Four is a two-player game. The discs of the first player are marked with X, and the discs of the second player are marked with O. The players take turns dropping their disc from the top into a *n*-column, *m*-row vertically suspended grid. The pieces fall straight down, occupying the lowest available space within the column. The objective of the game is to be the first to form a horizontal, vertical, or diagonal line of four of one's own discs. If the grid becomes full, the result is draw.

Implement this game, and let the grid size be selectable (8x5, 10x6, 12x7). The game should recognize if it is ended, and it has to show the name of the winner in a message box (if the game is not ended with draw), and automatically begin a new game.

2. short description how to use your program (user doc)

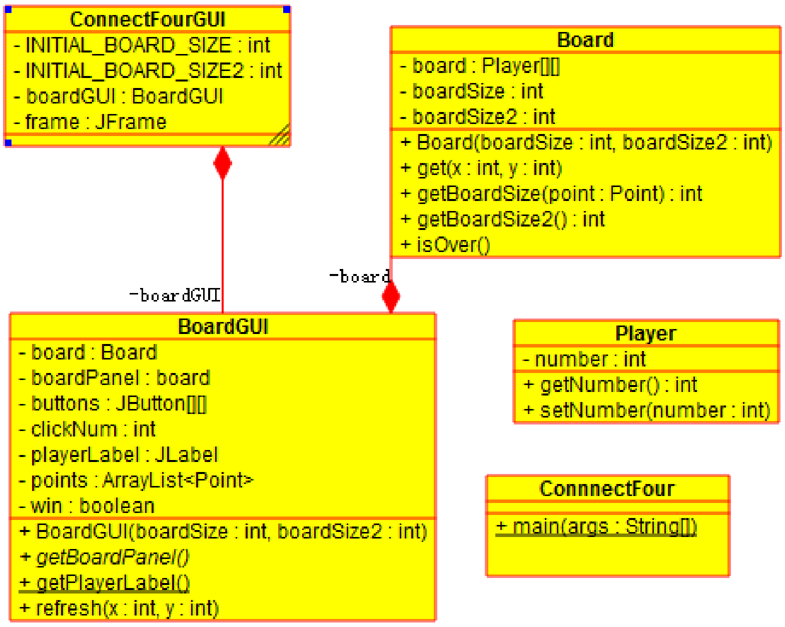
1. This game is called connect four for two players, which aim is to make the FIRST (winner) to form a horizontal, vertical, or diagonal line of Four of one's own discs.

2. The first player is marked with X in the game while the second player is marked with O in the game

3. Every time each player clicked on each grip, the disc will be dropped from the top into a *n*-column, *m*-row vertically suspended grid. The pieces fall straight down, occupying the lowest available space within the column.

4. If the grid becomes full with no winner, then result is draw.

3. UML class diagram (made with a dedicated UML tool)



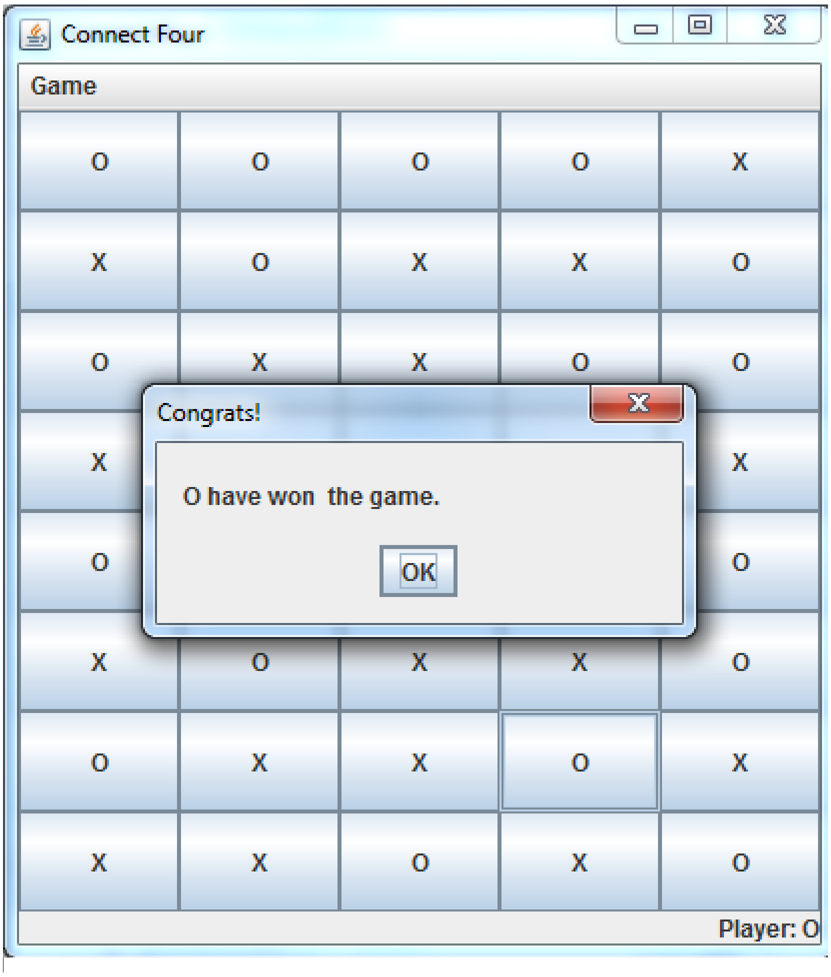
4. short descriptions of the implemented methods

```
public Player get(int x, int y) {                          // set every board for the game
            return board[x][y];
      }

public Player get(Point point) {                        // make every grid get access as a point
                  int x = (int)point.getX()
                  int y = (int)point.getY()
                  return get(x, y);
            }

public int getBoardSize() {
```

```
          return boardSize;                      // get access to the width of the board
}

public int getBoardSize()
              return boardSize;              // get access to the length of the board
}

public Board(int boardSize,int boardSize2)    // set the board

public JPanel getBoardPanel(){         // get access to the boardPanel
          return boardPanel;
}

public void refresh(int x, int y)      // the main function for the action of every point
```

1. • list of test cases you have tested (at least 10 pieces)

1.



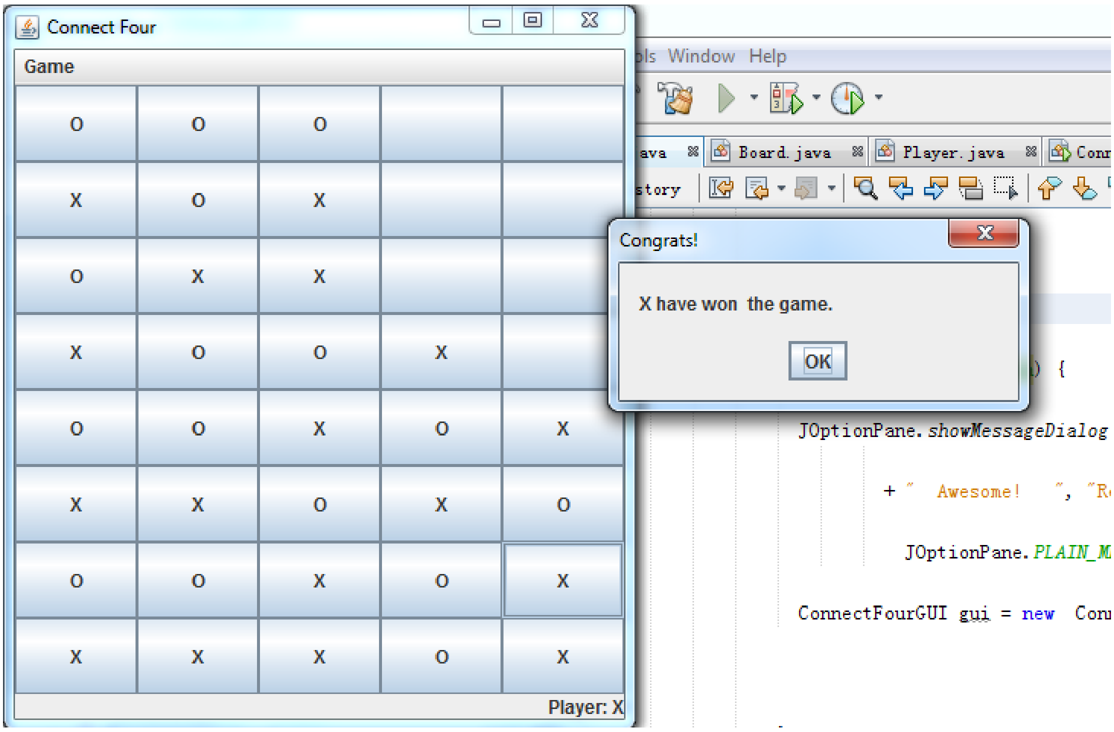
O	O	O	O	X
X	O	X	X	O
O	X	X	O	O
X				X
O				O
X	O	X	X	O
O	X	X	O	X
X	X	O	X	O

Player: O

Congrats!

O have won the game.

2.



O	O	O		
X	O	X		
O	X	X		
X	O	O	X	
O	O	X	O	X
X	X	O	X	O
O	O	X	O	X
X	X	X	O	X

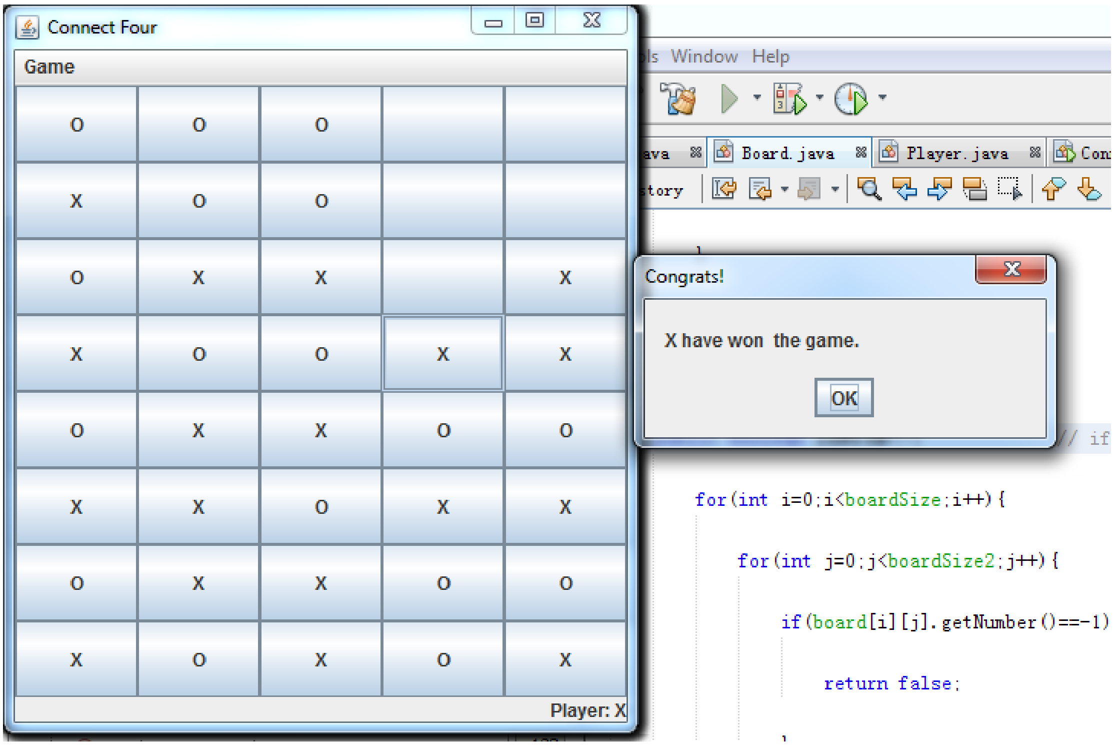
Player: X

Congrats!

X have won the game.

```
                                    ) {
            JOptionPane. showMessageDialog
                    + "  Awesome!  ", "R
                    JOptionPane. PLAIN_M
            ConnectFourGUI gui = new Con
```

3.

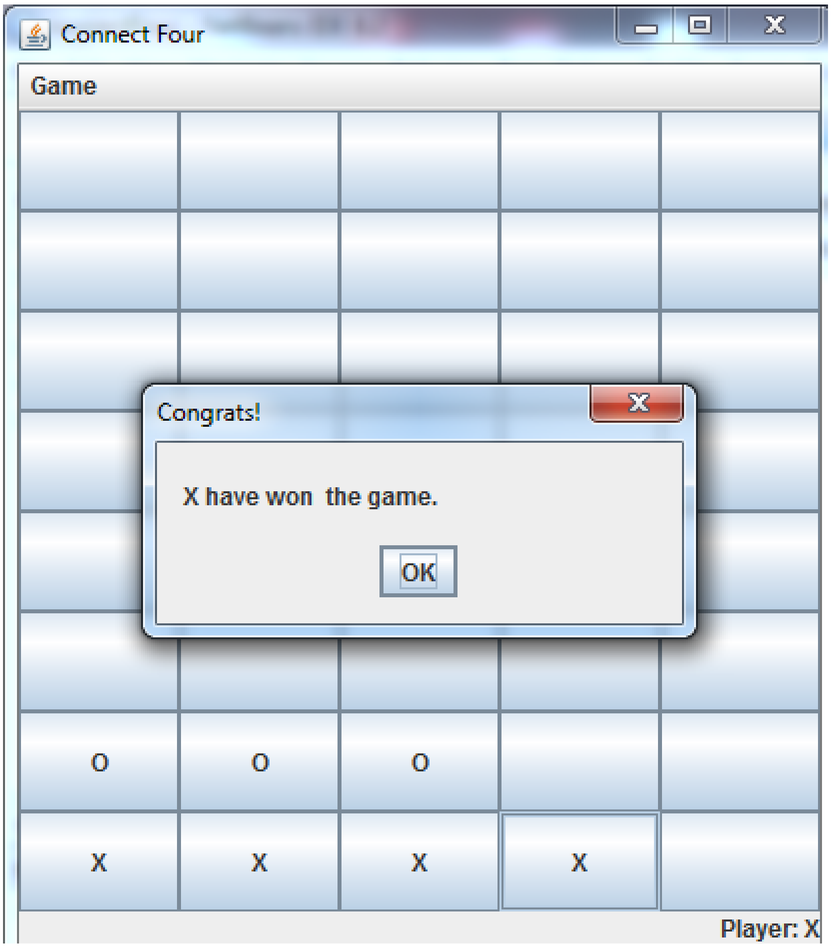


O	O	O		
X	O	O		
O	X	X		X
X	O	O	X	X
O	X	X	O	O
X	X	O	X	X
O	X	X	O	O
X	O	X	O	X

Congrats!

X have won the game.

```
// if
for(int i=0;i<boardSize;i++){
    for(int j=0;j<boardSize2;j++){
        if(board[i][j].getNumber()==-1)
            return false;
```



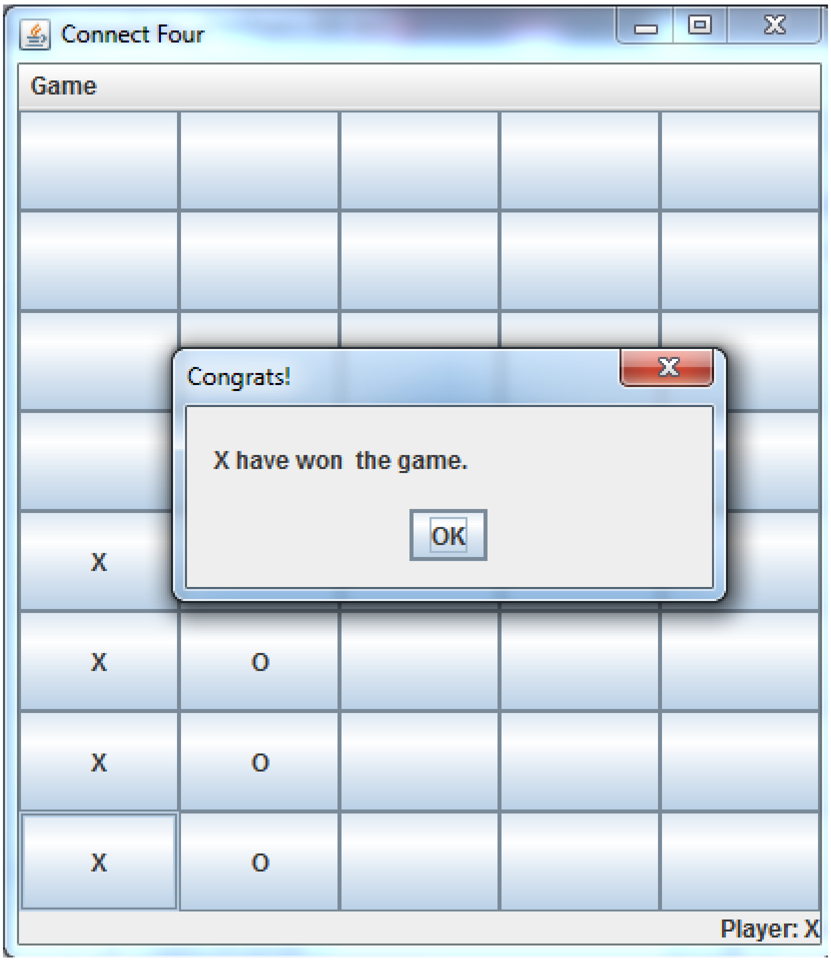
O	O	O		
X	X	X	X	

Congrats!

X have won the game.

4.

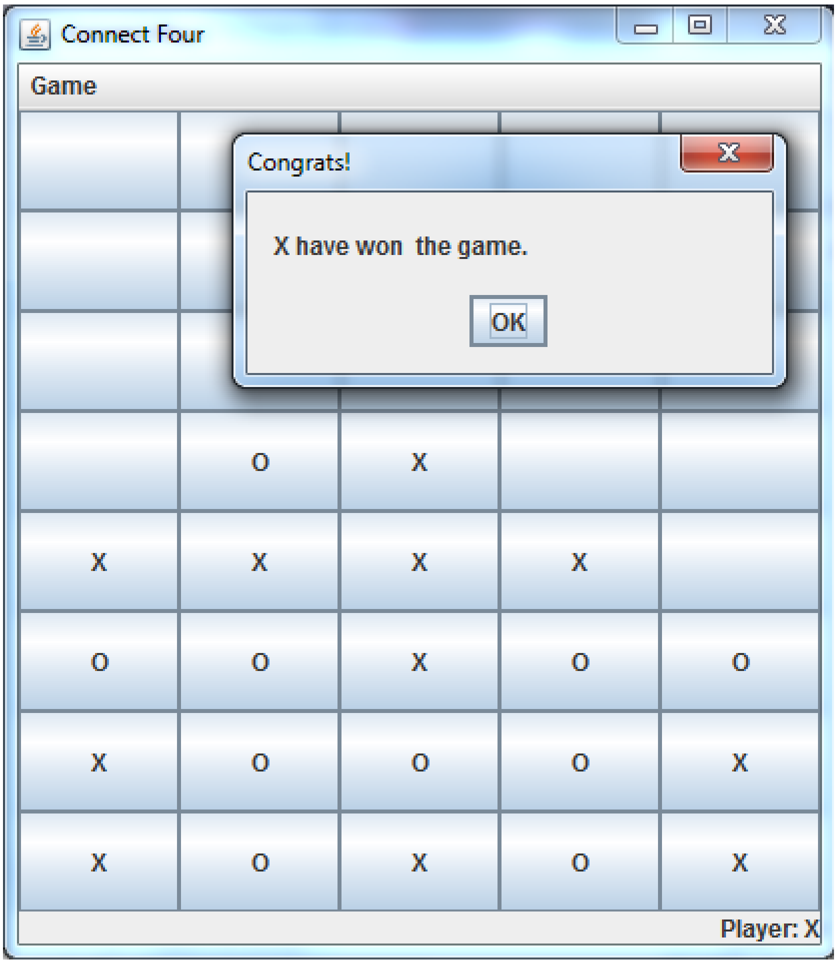
5.



Congrats!

X have won the game.

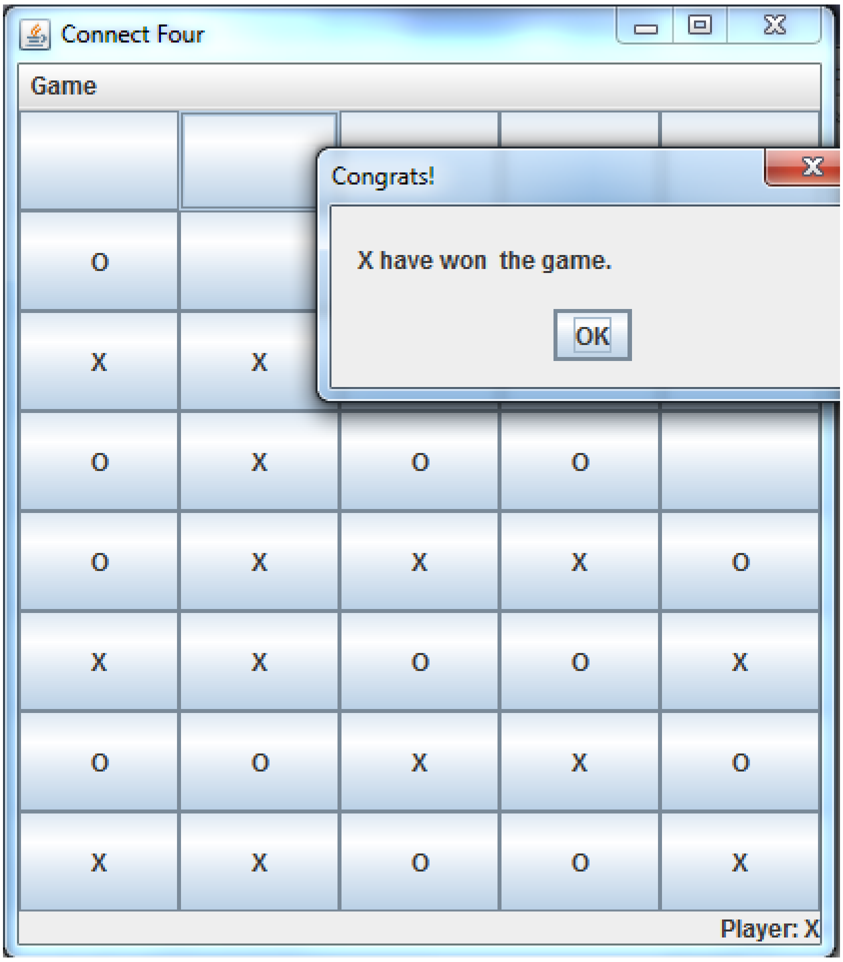
6.



X have won the game.

	O	X		
X	X	X	X	
O	O	X	O	O
X	O	O	O	X
X	O	X	O	X

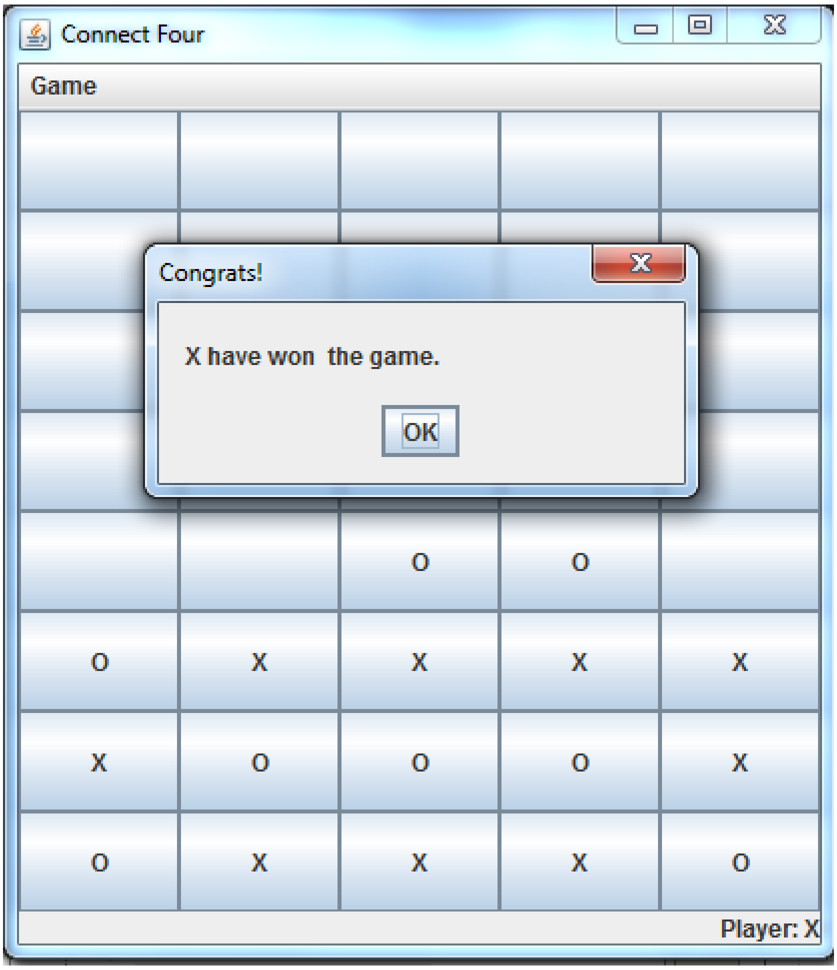
7.



X have won the game.

O				
X	X			
O	X	O	O	
O	X	X	X	O
X	X	O	O	X
O	O	X	X	O
X	X	O	O	X

8.



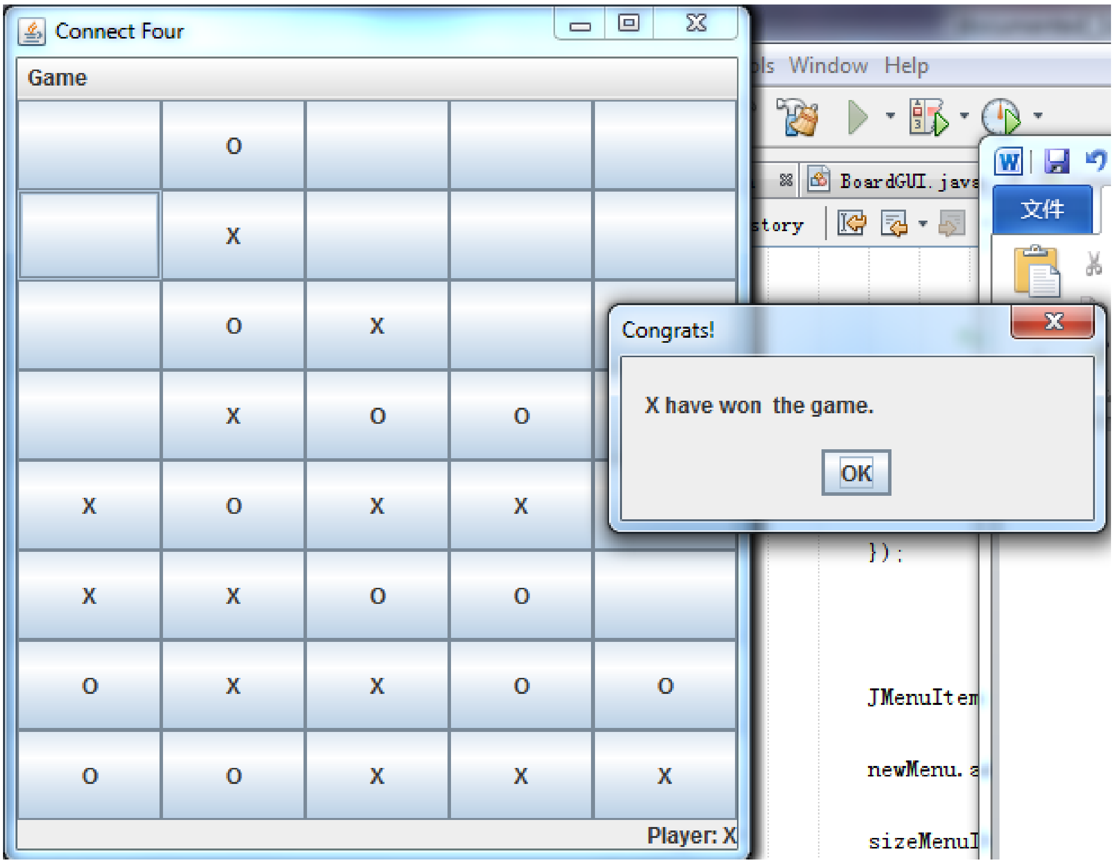
Congrats!

X have won the game.

		O	O	
O	X	X	X	X
X	O	O	O	X
O	X	X	X	O

Player: X

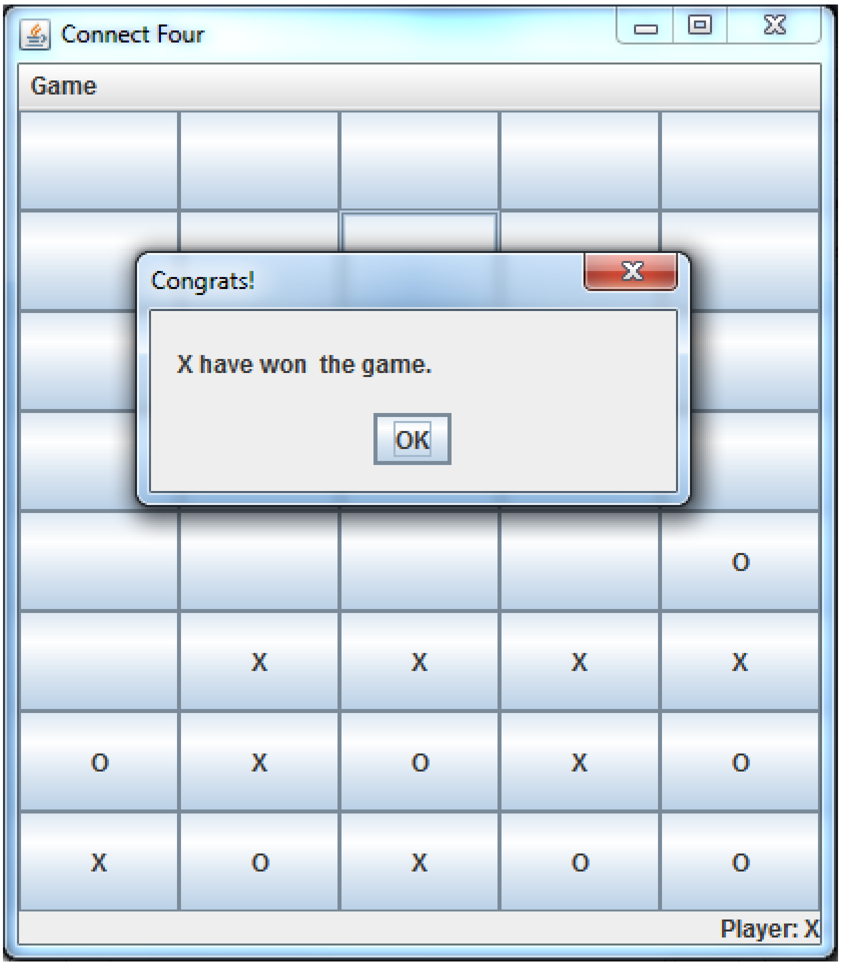
9.



	O			
	X			
	O	X		
	X	O	O	
X	O	X	X	
X	X	O	O	
O	X	X	O	O
O	O	X	X	X

X have won the game.

10.



Congrats!

X have won the game.

				O
	X	X	X	X
O	X	O	X	O
X	O	X	O	O

Player: X