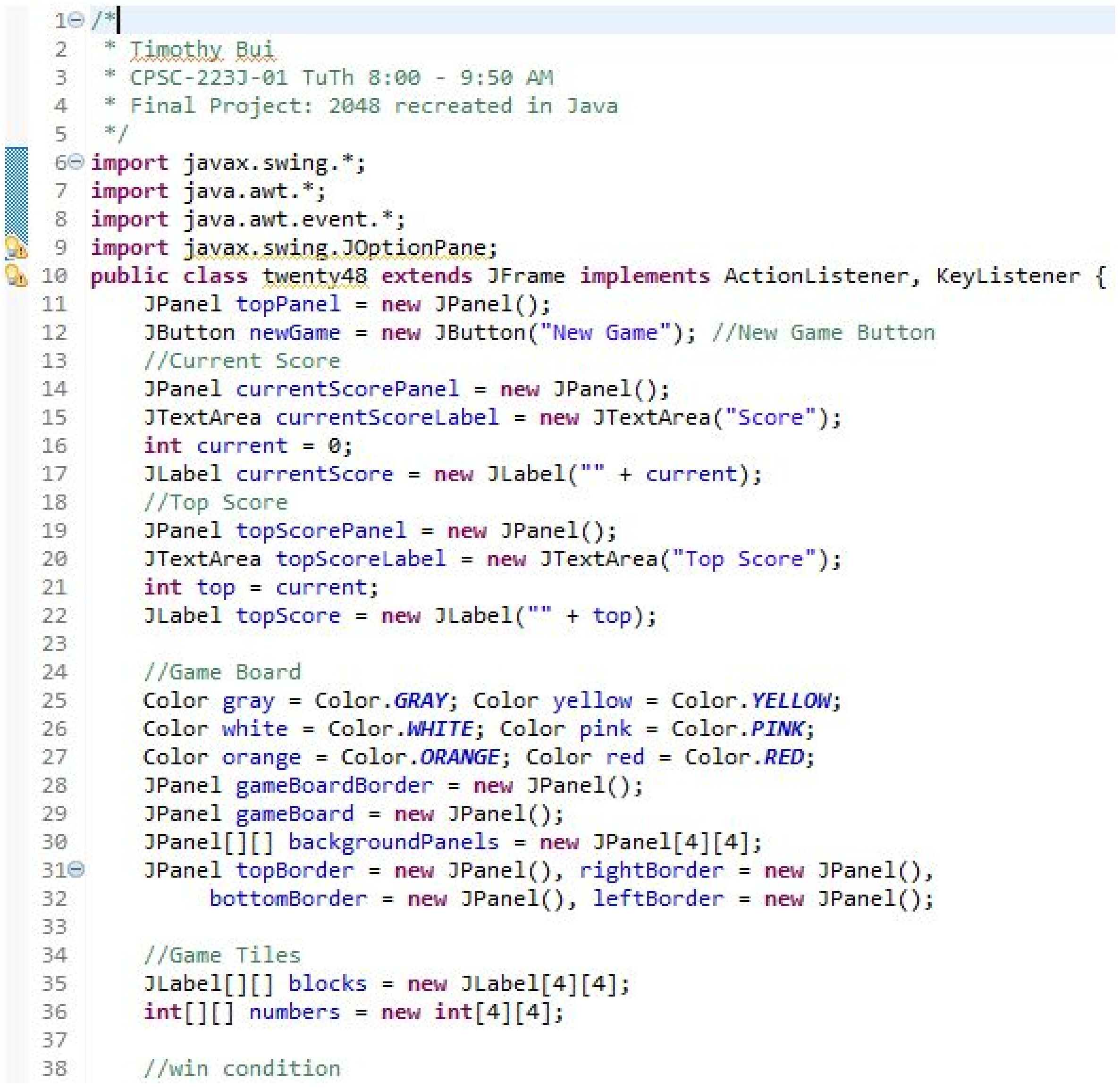
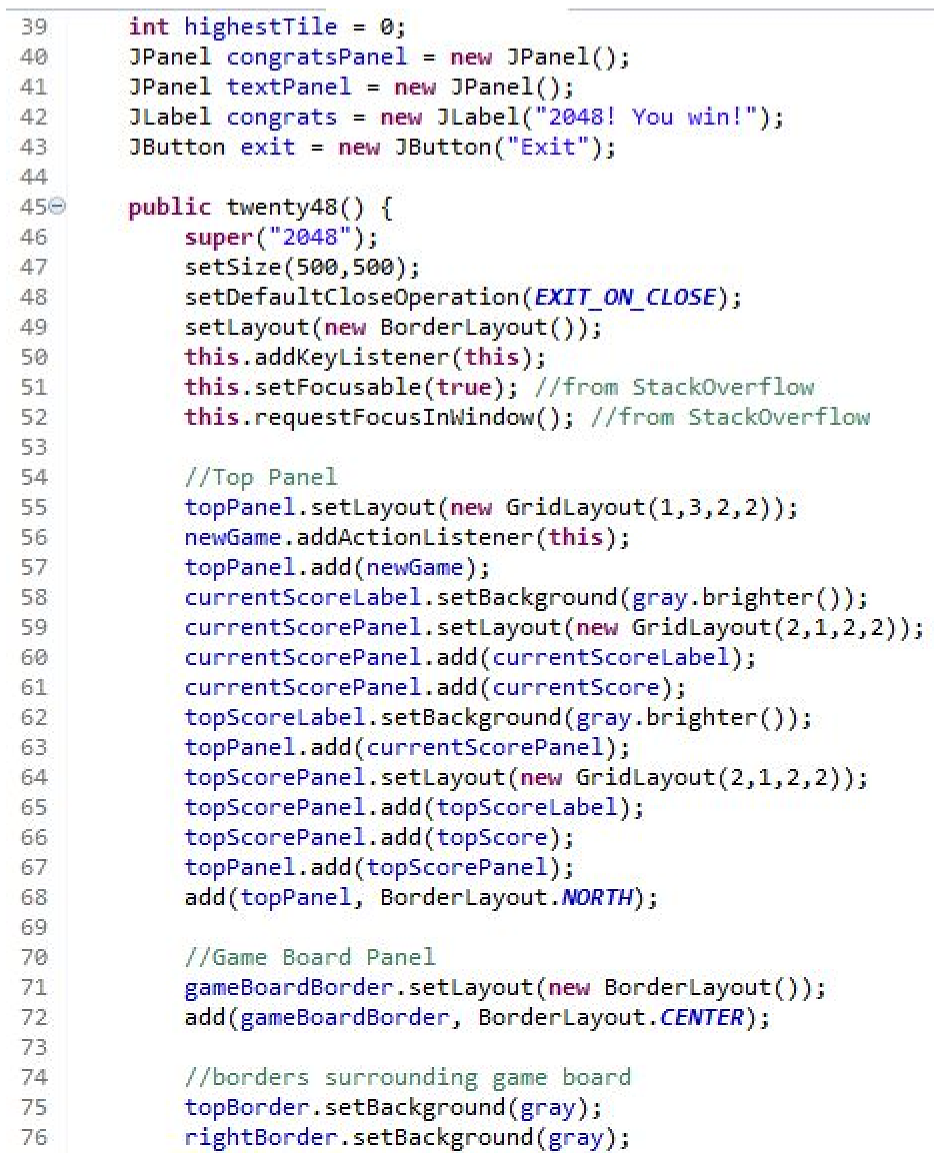
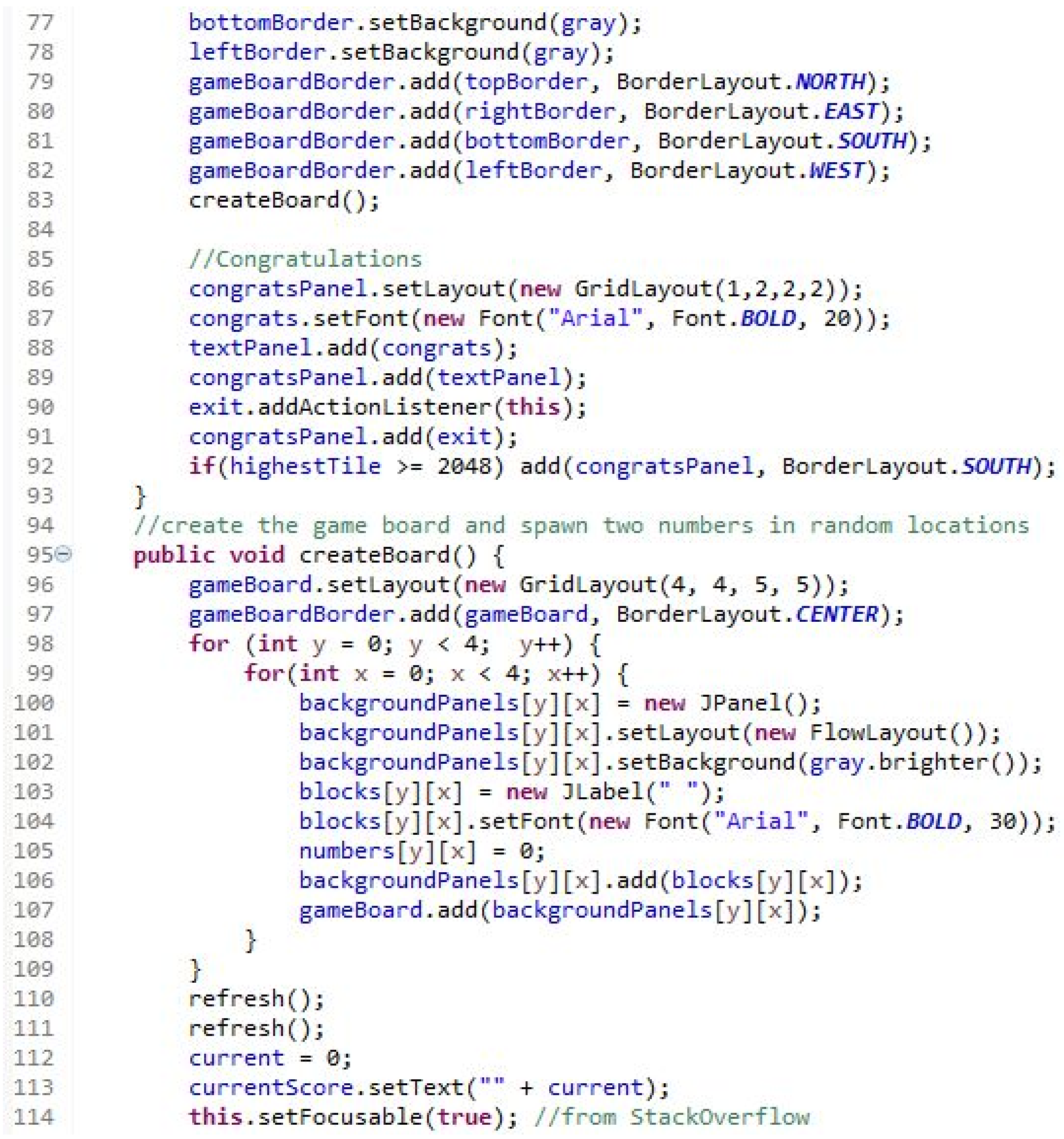


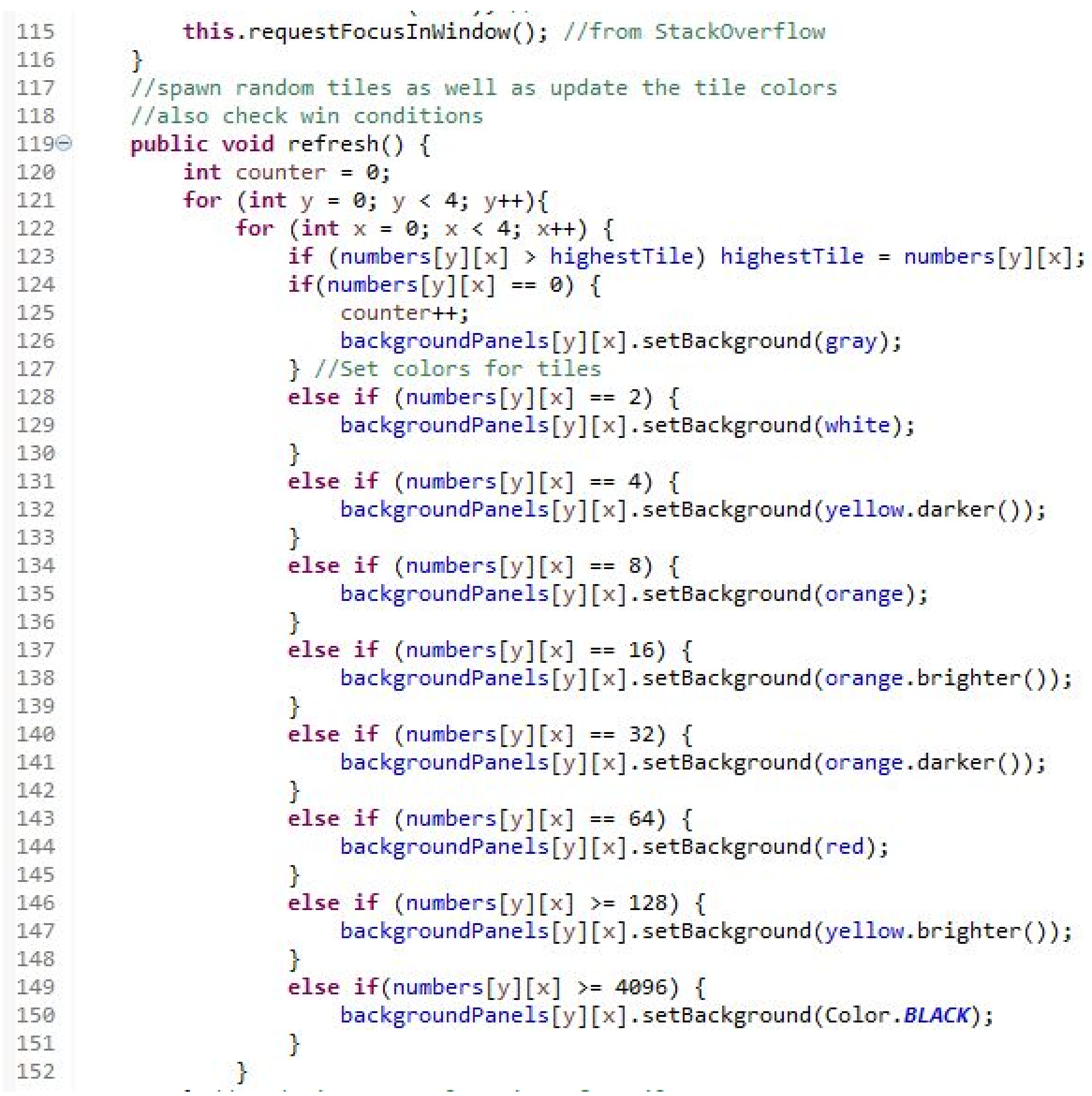
```java
/*
 * Timothy Bui
 * CPSC-223J-01 TuTh 8:00 - 9:50 AM
 * Final Project: 2048 recreated in Java
 */
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.JOptionPane;
public class twenty48 extends JFrame implements ActionListener, KeyListener {
    JPanel topPanel = new JPanel();
    JButton newGame = new JButton("New Game"); //New Game Button
    //Current Score
    JPanel currentScorePanel = new JPanel();
    JTextArea currentScoreLabel = new JTextArea("Score");
    int current = 0;
    JLabel currentScore = new JLabel("" + current);
    //Top Score
    JPanel topScorePanel = new JPanel();
    JTextArea topScoreLabel = new JTextArea("Top Score");
    int top = current;
    JLabel topScore = new JLabel("" + top);

    //Game Board
    Color gray = Color.GRAY; Color yellow = Color.YELLOW;
    Color white = Color.WHITE; Color pink = Color.PINK;
    Color orange = Color.ORANGE; Color red = Color.RED;
    JPanel gameBoardBorder = new JPanel();
    JPanel gameBoard = new JPanel();
    JPanel[][] backgroundPanels = new JPanel[4][4];
    JPanel topBorder = new JPanel(), rightBorder = new JPanel(),
         bottomBorder = new JPanel(), leftBorder = new JPanel();

    //Game Tiles
    JLabel[][] blocks = new JLabel[4][4];
    int[][] numbers = new int[4][4];

    //win condition
```



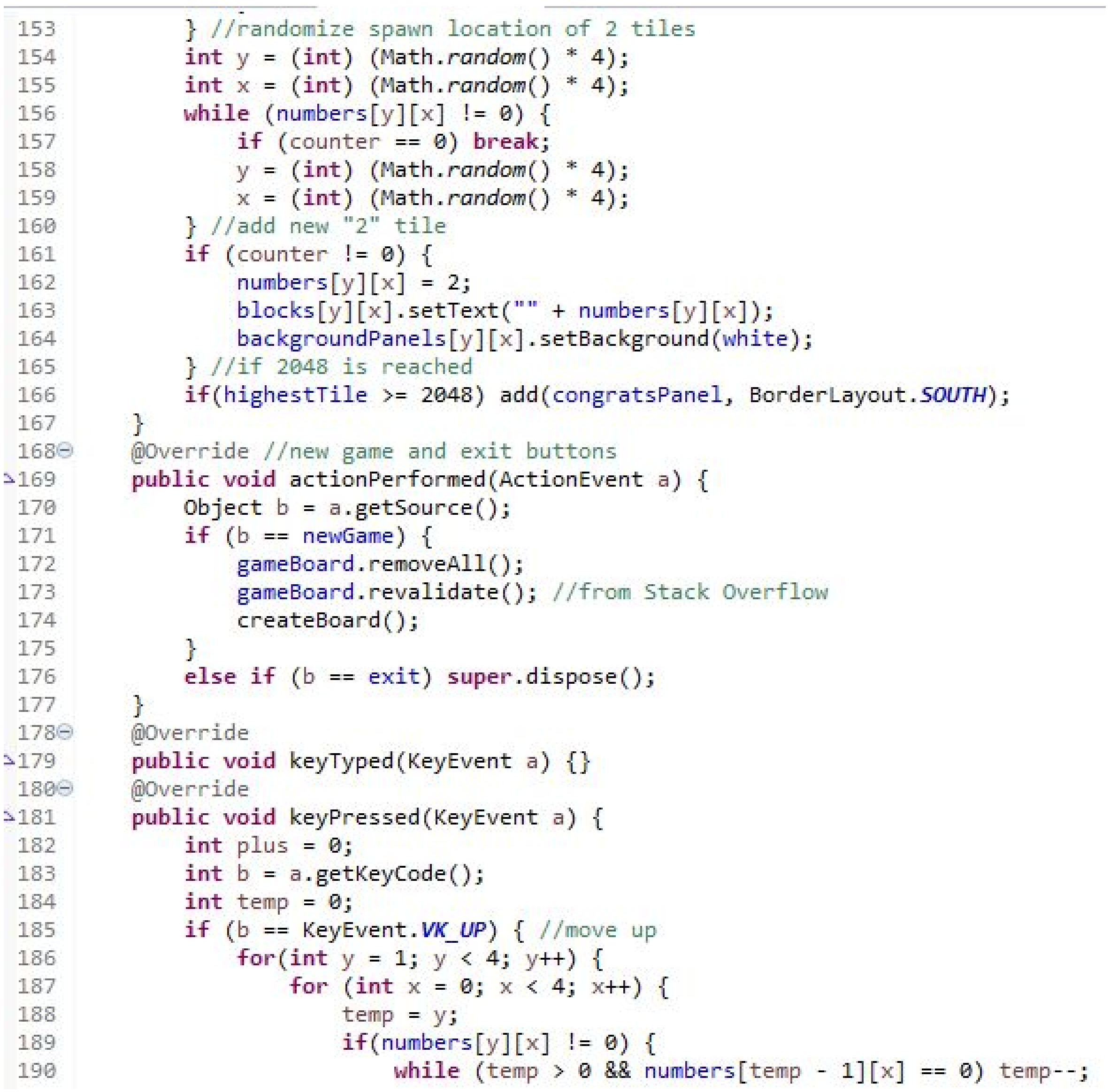
```java
    int highestTile = 0;
    JPanel congratsPanel = new JPanel();
    JPanel textPanel = new JPanel();
    JLabel congrats = new JLabel("2048! You win!");
    JButton exit = new JButton("Exit");

    public twenty48() {
        super("2048");
        setSize(500,500);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
        this.addKeyListener(this);
        this.setFocusable(true); //from StackOverflow
        this.requestFocusInWindow(); //from StackOverflow

        //Top Panel
        topPanel.setLayout(new GridLayout(1,3,2,2));
        newGame.addActionListener(this);
        topPanel.add(newGame);
        currentScoreLabel.setBackground(gray.brighter());
        currentScorePanel.setLayout(new GridLayout(2,1,2,2));
        currentScorePanel.add(currentScoreLabel);
        currentScorePanel.add(currentScore);
        topScoreLabel.setBackground(gray.brighter());
        topPanel.add(currentScorePanel);
        topScorePanel.setLayout(new GridLayout(2,1,2,2));
        topScorePanel.add(topScoreLabel);
        topScorePanel.add(topScore);
        topPanel.add(topScorePanel);
        add(topPanel, BorderLayout.NORTH);

        //Game Board Panel
        gameBoardBorder.setLayout(new BorderLayout());
        add(gameBoardBorder, BorderLayout.CENTER);

        //borders surrounding game board
        topBorder.setBackground(gray);
        rightBorder.setBackground(gray);
```



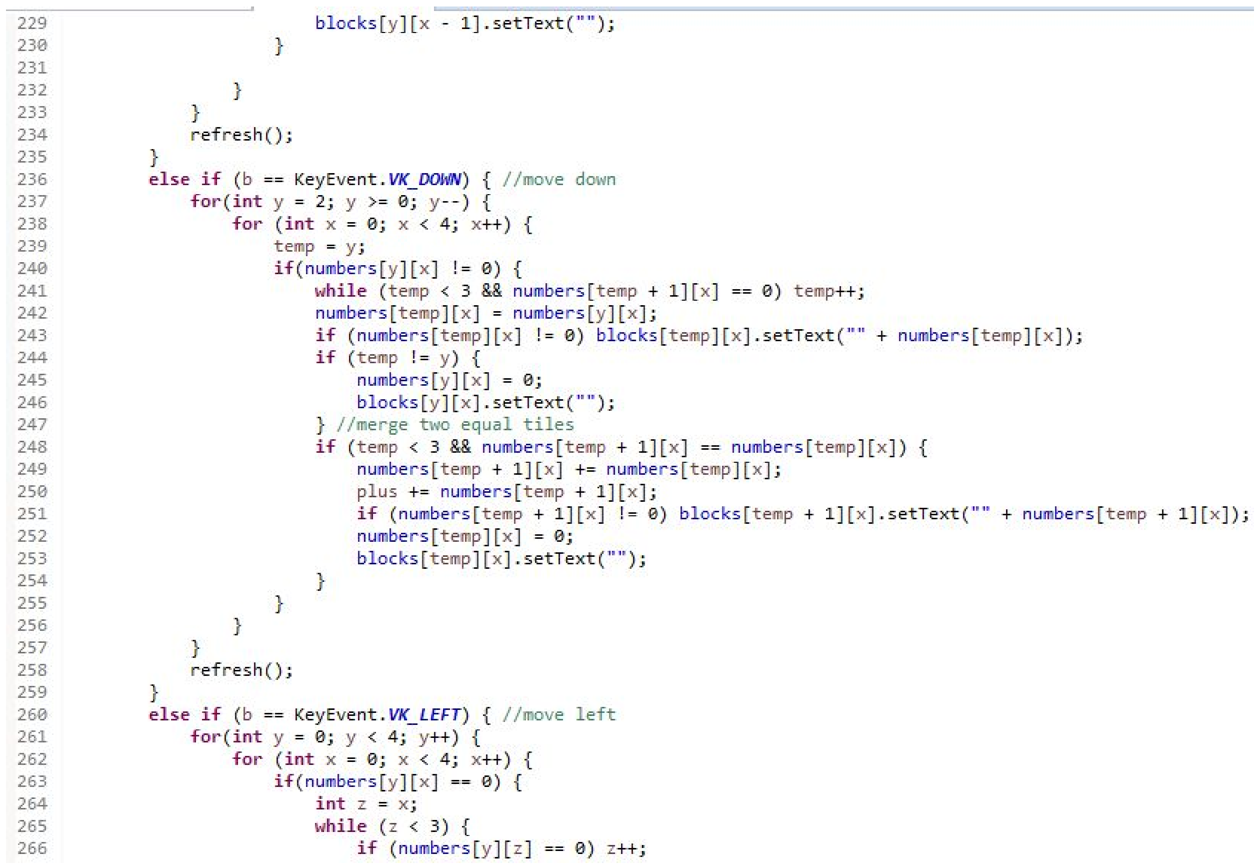
```java
        bottomBorder.setBackground(gray);
        leftBorder.setBackground(gray);
        gameBoardBorder.add(topBorder, BorderLayout.NORTH);
        gameBoardBorder.add(rightBorder, BorderLayout.EAST);
        gameBoardBorder.add(bottomBorder, BorderLayout.SOUTH);
        gameBoardBorder.add(leftBorder, BorderLayout.WEST);
        createBoard();

        //Congratulations
        congratsPanel.setLayout(new GridLayout(1,2,2,2));
        congrats.setFont(new Font("Arial", Font.BOLD, 20));
        textPanel.add(congrats);
        congratsPanel.add(textPanel);
        exit.addActionListener(this);
        congratsPanel.add(exit);
        if(highestTile >= 2048) add(congratsPanel, BorderLayout.SOUTH);
    }
    //create the game board and spawn two numbers in random locations
    public void createBoard() {
        gameBoard.setLayout(new GridLayout(4, 4, 5, 5));
        gameBoardBorder.add(gameBoard, BorderLayout.CENTER);
        for (int y = 0; y < 4;  y++) {
            for(int x = 0; x < 4; x++) {
                backgroundPanels[y][x] = new JPanel();
                backgroundPanels[y][x].setLayout(new FlowLayout());
                backgroundPanels[y][x].setBackground(gray.brighter());
                blocks[y][x] = new JLabel(" ");
                blocks[y][x].setFont(new Font("Arial", Font.BOLD, 30));
                numbers[y][x] = 0;
                backgroundPanels[y][x].add(blocks[y][x]);
                gameBoard.add(backgroundPanels[y][x]);
            }
        }
        refresh();
        refresh();
        current = 0;
        currentScore.setText("" + current);
        this.setFocusable(true); //from StackOverflow
```



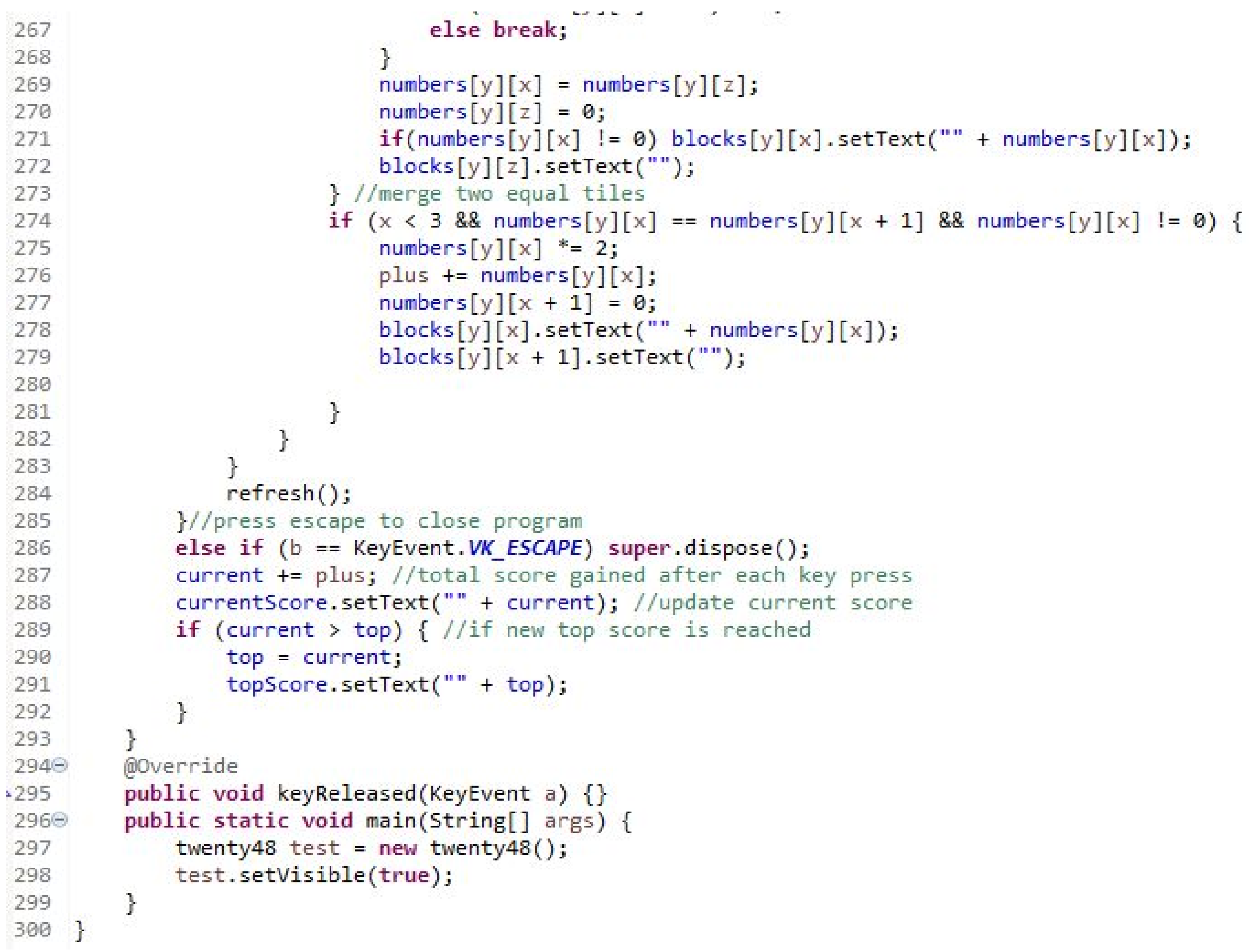
```java
        this.requestFocusInWindow(); //from StackOverflow
    }
    //spawn random tiles as well as update the tile colors
    //also check win conditions
    public void refresh() {
        int counter = 0;
        for (int y = 0; y < 4; y++){
            for (int x = 0; x < 4; x++) {
                if (numbers[y][x] > highestTile) highestTile = numbers[y][x];
                if(numbers[y][x] == 0) {
                    counter++;
                    backgroundPanels[y][x].setBackground(gray);
                } //Set colors for tiles
                else if (numbers[y][x] == 2) {
                    backgroundPanels[y][x].setBackground(white);
                }
                else if (numbers[y][x] == 4) {
                    backgroundPanels[y][x].setBackground(yellow.darker());
                }
                else if (numbers[y][x] == 8) {
                    backgroundPanels[y][x].setBackground(orange);
                }
                else if (numbers[y][x] == 16) {
                    backgroundPanels[y][x].setBackground(orange.brighter());
                }
                else if (numbers[y][x] == 32) {
                    backgroundPanels[y][x].setBackground(orange.darker());
                }
                else if (numbers[y][x] == 64) {
                    backgroundPanels[y][x].setBackground(red);
                }
                else if (numbers[y][x] >= 128) {
                    backgroundPanels[y][x].setBackground(yellow.brighter());
                }
                else if(numbers[y][x] >= 4096) {
                    backgroundPanels[y][x].setBackground(Color.BLACK);
                }
            }
```



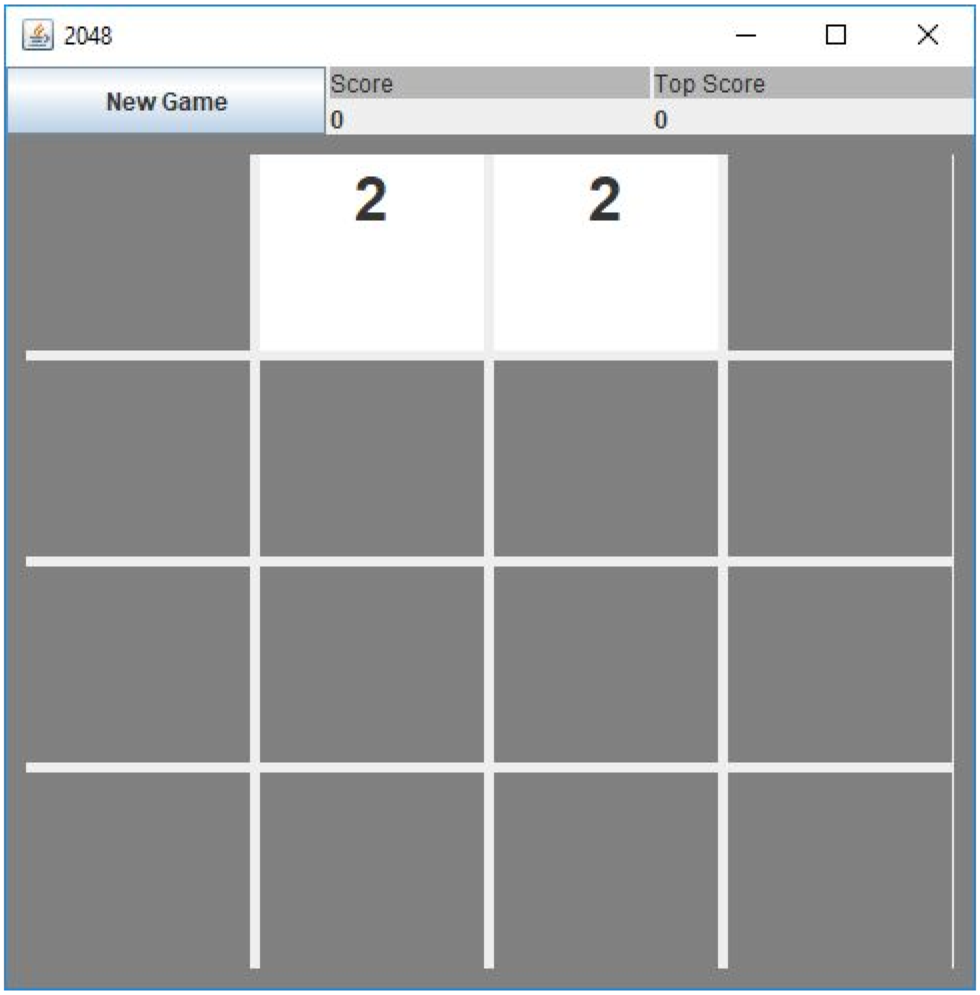
```java
        } //randomize spawn location of 2 tiles
        int y = (int) (Math.random() * 4);
        int x = (int) (Math.random() * 4);
        while (numbers[y][x] != 0) {
            if (counter == 0) break;
            y = (int) (Math.random() * 4);
            x = (int) (Math.random() * 4);
        } //add new "2" tile
        if (counter != 0) {
            numbers[y][x] = 2;
            blocks[y][x].setText("" + numbers[y][x]);
            backgroundPanels[y][x].setBackground(white);
        } //if 2048 is reached
        if(highestTile >= 2048) add(congratsPanel, BorderLayout.SOUTH);
    }
    @Override //new game and exit buttons
    public void actionPerformed(ActionEvent a) {
        Object b = a.getSource();
        if (b == newGame) {
            gameBoard.removeAll();
            gameBoard.revalidate(); //from Stack Overflow
            createBoard();
        }
        else if (b == exit) super.dispose();
    }
    @Override
    public void keyTyped(KeyEvent a) {}
    @Override
    public void keyPressed(KeyEvent a) {
        int plus = 0;
        int b = a.getKeyCode();
        int temp = 0;
        if (b == KeyEvent.VK_UP) { //move up
            for(int y = 1; y < 4; y++) {
                for (int x = 0; x < 4; x++) {
                    temp = y;
                    if(numbers[y][x] != 0) {
                        while (temp > 0 && numbers[temp - 1][x] == 0) temp--;
                        numbers[temp][x] = numbers[y][x];
                        if (numbers[temp][x] != 0) blocks[temp][x].setText("" + numbers[temp][x]);
                        if (temp != y) {
                            numbers[y][x] = 0;
                            blocks[y][x].setText("");
                        } //merge two equal tiles
                        if (temp > 0 && numbers[temp - 1][x] == numbers[temp][x]) {
                            numbers[temp - 1][x] += numbers[temp][x];
                            plus += numbers[temp - 1][x];
                            if (numbers[temp - 1][x] != 0) blocks[temp - 1][x].setText("" + numbers[temp - 1][x]);
                            numbers[temp][x] = 0;
                            blocks[temp][x].setText("");
                        }
                    }
                }
            }
            refresh();
        }
        else if (b == KeyEvent.VK_RIGHT) { //move right
            for(int y = 0; y < 4; y++) {
                for (int x = 3; x > 0; x--) {
                    //find the nearest occupied tile and move it to this empty tile
                    if(numbers[y][x] == 0) {
                        int z = x;
                        while (z > 0) {
                            if (numbers[y][z] == 0) z--;
                            else break;
                        }
                        numbers[y][x] = numbers[y][z];
                        numbers[y][z] = 0;
                        if(numbers[y][x] != 0) blocks[y][x].setText("" + numbers[y][x]);
                        blocks[y][z].setText("");
                    } //merge two equal tiles
                    if (numbers[y][x] == numbers[y][x - 1] && numbers[y][x] != 0) {
                        numbers[y][x] *= 2;
                        plus += numbers[y][x];
                        numbers[y][x - 1] = 0;
                        blocks[y][x].setText("" + numbers[y][x]);
```

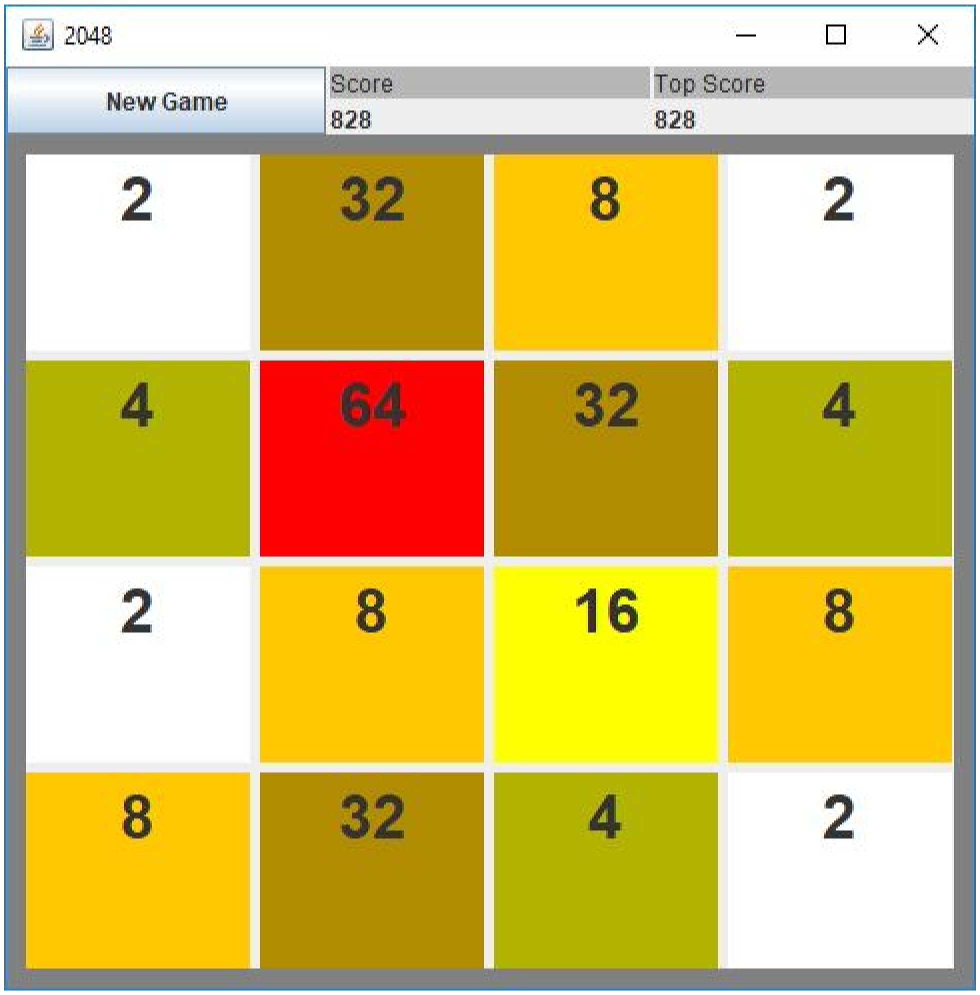


```java
                        blocks[y][x - 1].setText("");
                    }

                }
            }
            refresh();
        }
        else if (b == KeyEvent.VK_DOWN) { //move down
            for(int y = 2; y >= 0; y--) {
                for (int x = 0; x < 4; x++) {
                    temp = y;
                    if(numbers[y][x] != 0) {
                        while (temp < 3 && numbers[temp + 1][x] == 0) temp++;
                        numbers[temp][x] = numbers[y][x];
                        if (numbers[temp][x] != 0) blocks[temp][x].setText("" + numbers[temp][x]);
                        if (temp != y) {
                            numbers[y][x] = 0;
                            blocks[y][x].setText("");
                        } //merge two equal tiles
                        if (temp < 3 && numbers[temp + 1][x] == numbers[temp][x]) {
                            numbers[temp + 1][x] += numbers[temp][x];
                            plus += numbers[temp + 1][x];
                            if (numbers[temp + 1][x] != 0) blocks[temp + 1][x].setText("" + numbers[temp + 1][x]);
                            numbers[temp][x] = 0;
                            blocks[temp][x].setText("");
                        }
                    }
                }
            }
            refresh();
        }
        else if (b == KeyEvent.VK_LEFT) { //move left
            for(int y = 0; y < 4; y++) {
                for (int x = 0; x < 4; x++) {
                    if(numbers[y][x] == 0) {
                        int z = x;
                        while (z < 3) {
                            if (numbers[y][z] == 0) z++;
```



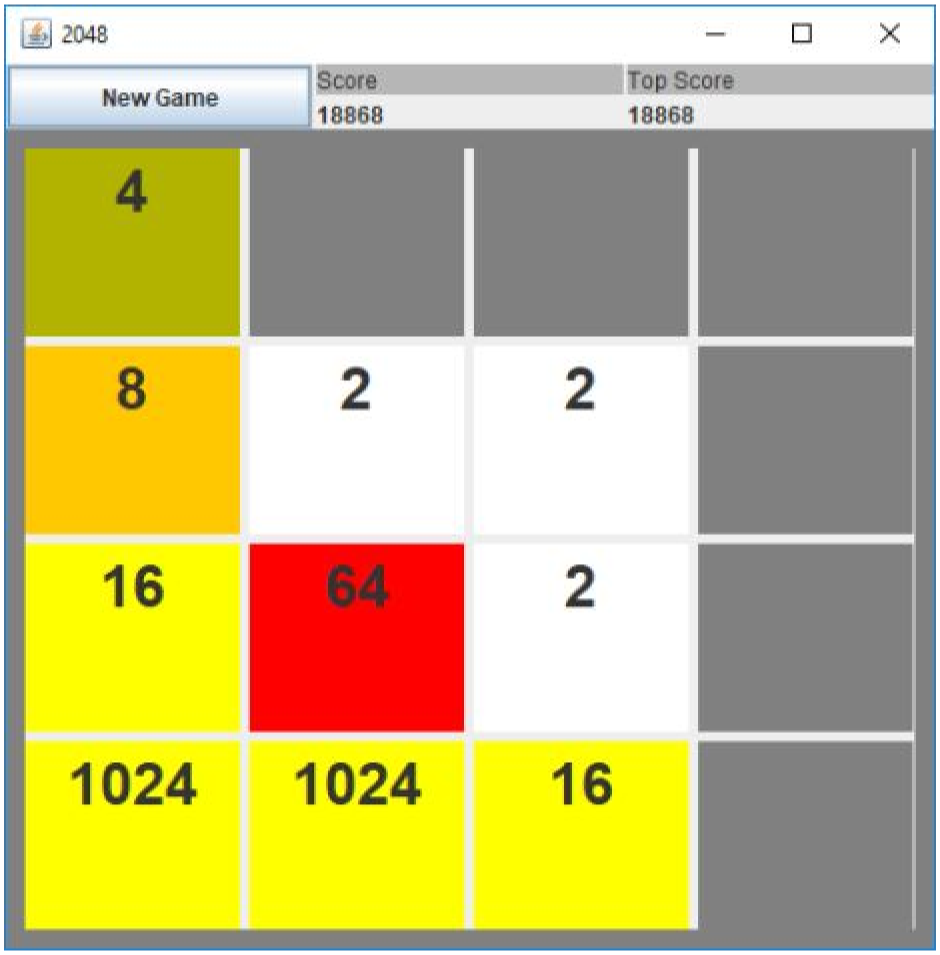
```java
                            else break;
                        }
                        numbers[y][x] = numbers[y][z];
                        numbers[y][z] = 0;
                        if(numbers[y][x] != 0) blocks[y][x].setText("" + numbers[y][x]);
                        blocks[y][z].setText("");
                    } //merge two equal tiles
                    if (x < 3 && numbers[y][x] == numbers[y][x + 1] && numbers[y][x] != 0) {
                        numbers[y][x] *= 2;
                        plus += numbers[y][x];
                        numbers[y][x + 1] = 0;
                        blocks[y][x].setText("" + numbers[y][x]);
                        blocks[y][x + 1].setText("");

                    }
                }
            }
            refresh();
        }//press escape to close program
        else if (b == KeyEvent.VK_ESCAPE) super.dispose();
        current += plus; //total score gained after each key press
        currentScore.setText("" + current); //update current score
        if (current > top) { //if new top score is reached
            top = current;
            topScore.setText("" + top);
        }
    }
    @Override
    public void keyReleased(KeyEvent a) {}
    public static void main(String[] args) {
        twenty48 test = new twenty48();
        test.setVisible(true);
    }
}
```


2048
New Game
Score
0
Top Score
0
2
2



Score: 828
Top Score: 828

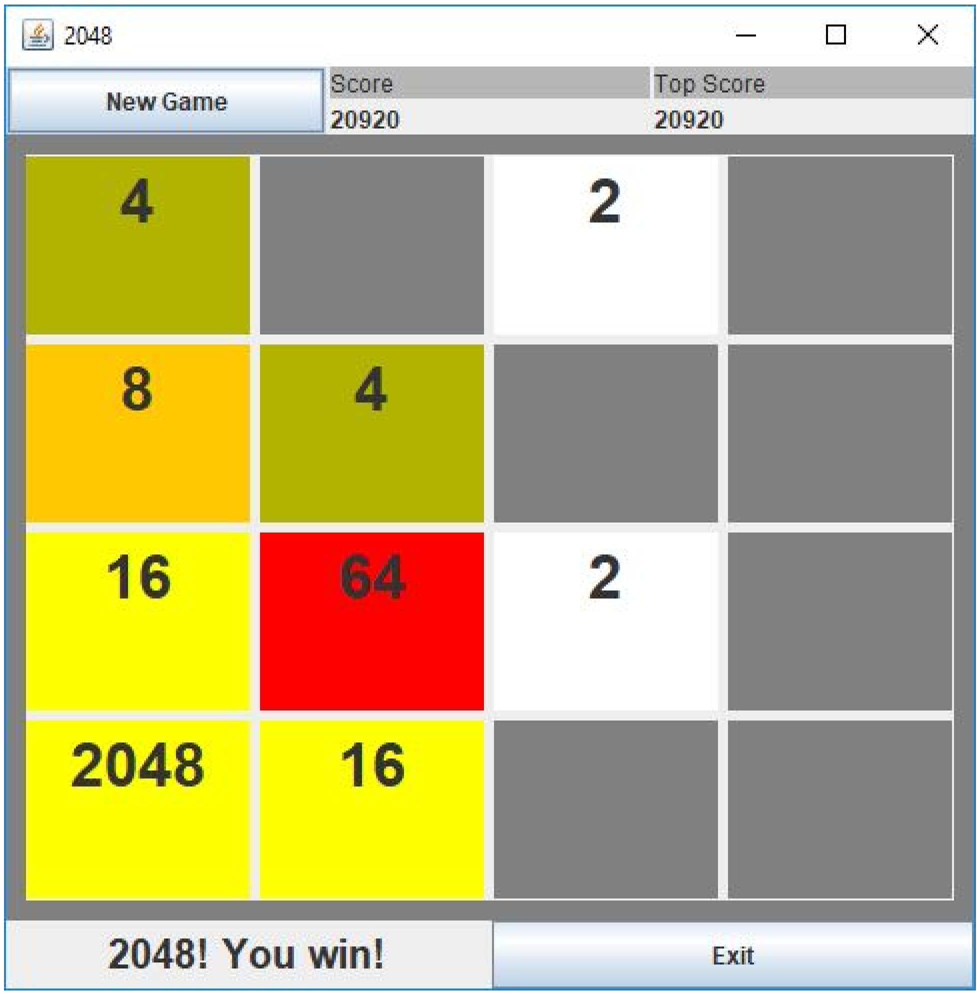
2	32	8	2
4	64	32	4
2	8	16	8
8	32	4	2



Score
18868

Top Score
18868

4			
8	2	2	
16	64	2	
1024	1024	16	



Score
20920

Top Score
20920

4		2	
8	4		
16	64	2	
2048	16		

2048! You win!