MEF UNIVERSITY

YOUR FREEDOM IN LEARNING

PROJECT 2

TETRIS 2048

COMP 204 - Programming Studio

Instructor: Prof.Dr. Muhittin Gökmen

Rabia Kodal - 041701018
Mert Burma - 041701033
Safak Barıs - 041701007

Date: 12.04.2020

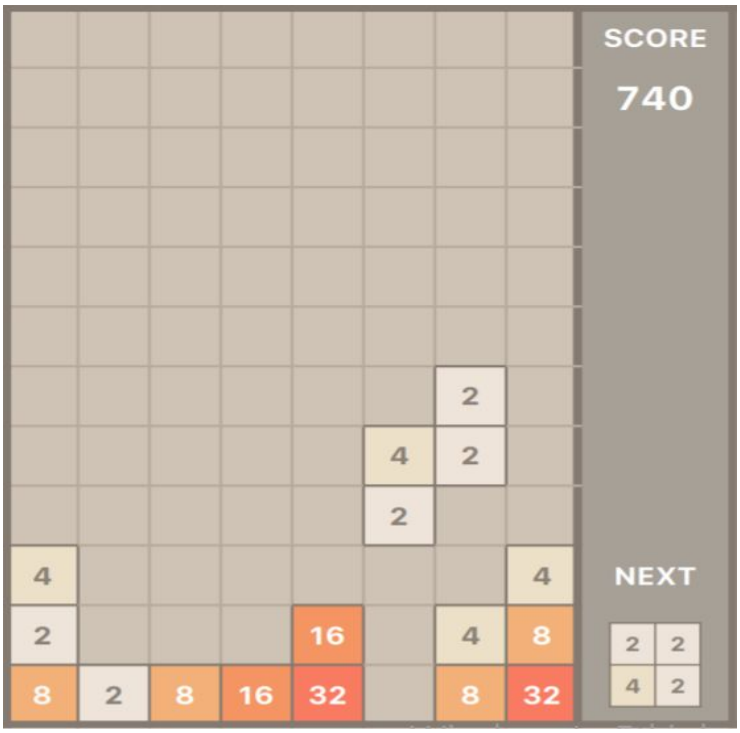
Abstract

The idea behind this project is to implement, review and combine the features of the popular video game Tetris and the 2048 game. The game is a combination of two well known games: Tetris and 2048 (https://www.youtube.com/watch?v=hWAYuv-2FwM is a sample video showing a version of the game).

Introduction

As a first step in this project we will develop the basic Tetris game. http://zetcode.com/tutorials/javagamestutorial/tetris/ is a tutorial showing step-by-step creation of a tetris game in Java.

Then you will combine it with 2048. https://rosettacode.org/wiki/2048 is a webpage that contains implementations of 2048 in a variety of programming languages.



An example image for the game is shown above.

The game has colors like the 2048 game and consists of standard tetris shapes (O, I, Z, S, T, L, J) and score and the incoming shape(net piece) are shown on the right side.

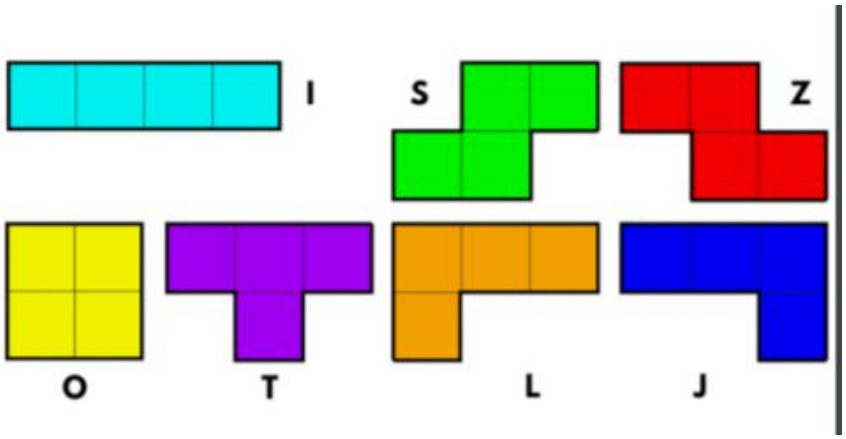
Description of the project

About Tetris:

The goal of Tetris is to clear as many horizontal block lines as possible. When the line is completely filled with tetrominos, the game will clear the horizontal block that

is full. If more than one line is filled at the same time, all horizontal blocks that are filled simultaneously will be cleared.

The player can move, rotate and drop different shapes, each consisting of four blocks (called tetromino and shown in the picture below).

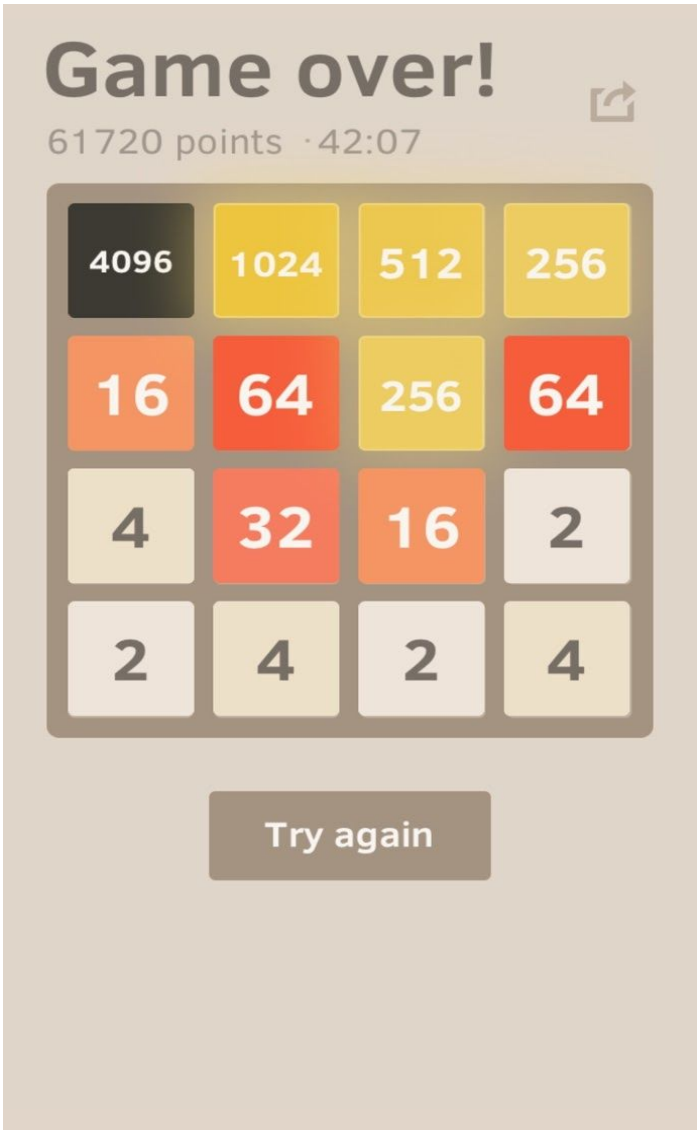


About 2048:

2048 is a small puzzle game in which the player moves tiles on a 4x4 board.

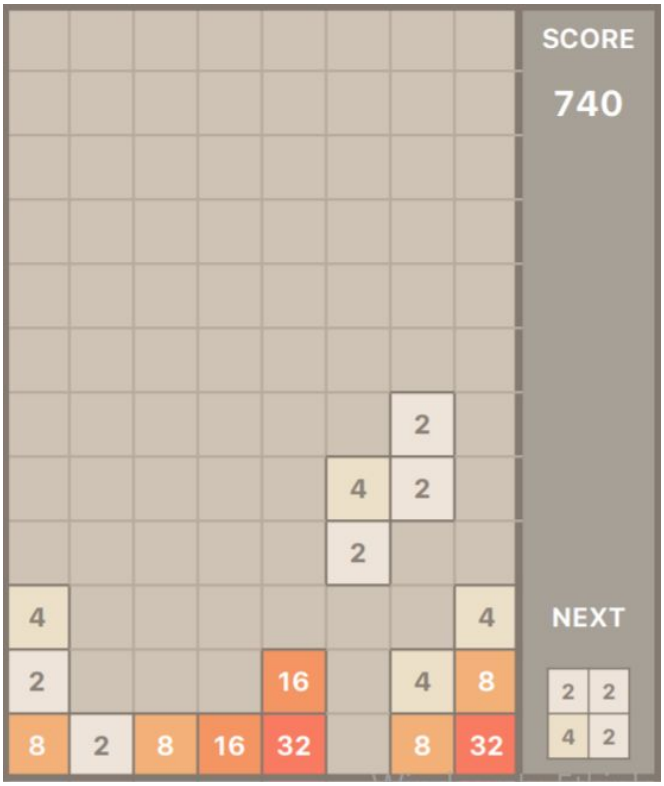
When moving tiles, if two tiles of the same value are touched, their value doubles and merges into a single tile (that is, they add two tiles of the same value and show them as new values). For each move, a random tile with a random value of 2 or 4 is created in the 4x4 board (You can also add the number 8 to make the game difficult if you wish).The winning criteria are to reach 2048, but the game allows to continue after reaching 2048.If the entire 4x4 board fills up with tiles and no further moves are possible, the player loses the game.

As seen in the image below, the player can exceed 2048 and continue the game, and the game space to move.

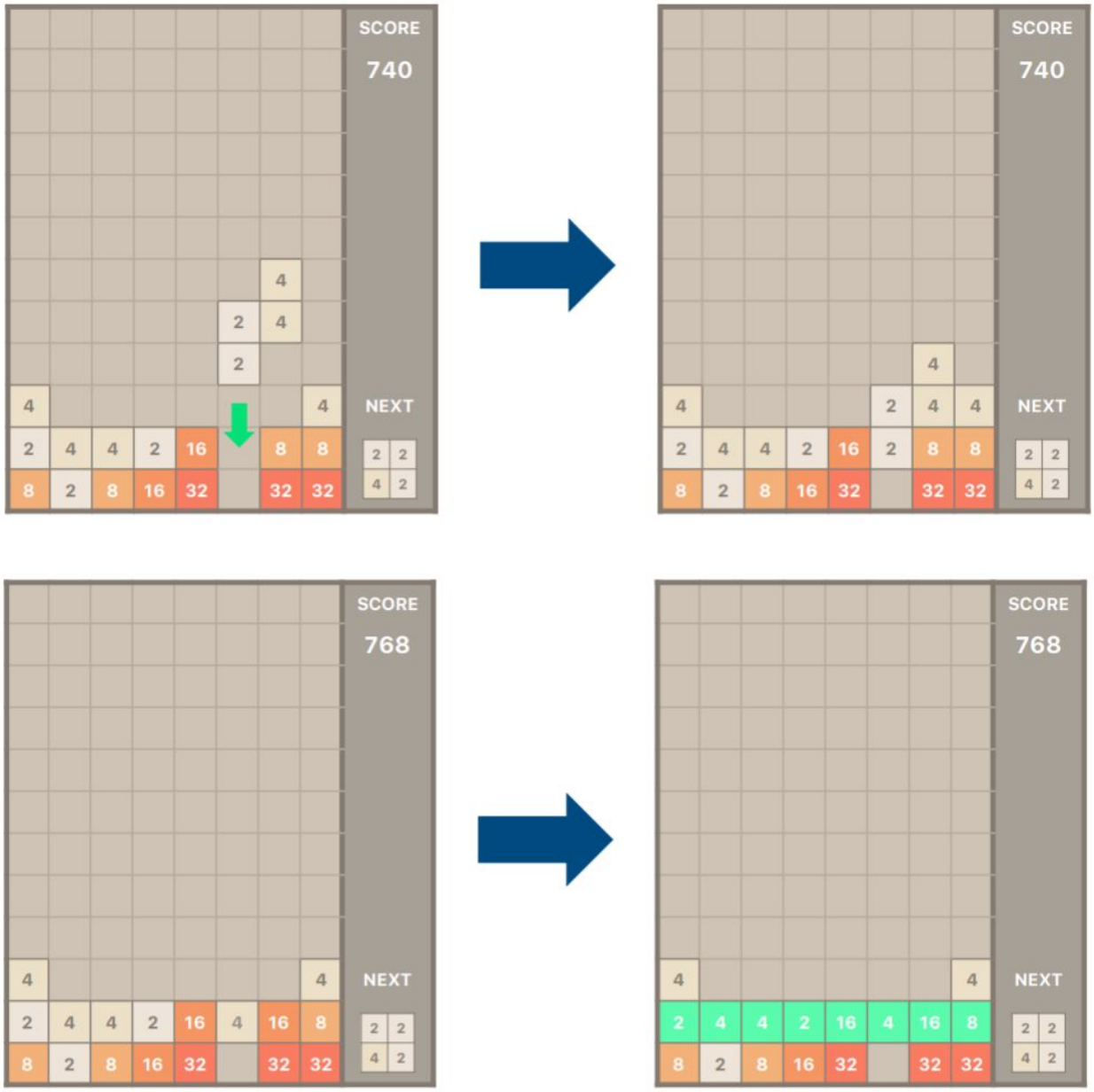


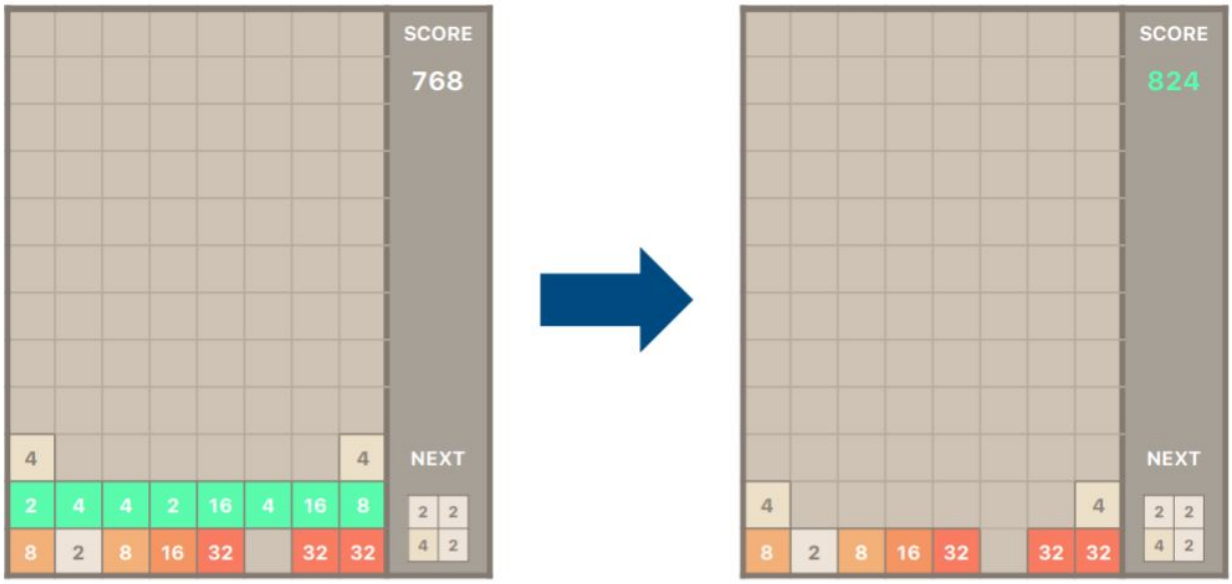
About Tetris 2048

Game play is similar to Tetris as the player can move and rotate different shapes each composed of 4 square tiles (tetrominoes shown below) in the 2D playing field (grid). The differences are the numbers (positive forces of 2) on the tiles that make up the tetrominos and the color determination according to the number on each tile .Each incoming tetromino consists of four tiles randomly numbered using either 2 or 4 (as in 2048).



Score is determined both from merged numbers as in 2048 and cleared horizontal rows as in Tetris.But first, the numbers that are the same blocks like in 2048 are combined and added to the scoring, then if there is a filled row, it is deleted and added to the scoring.Merging is performed from top to bottom.The most important detail is that combining numbers has a higher priority than clearing lines.





Description of my solution

First we need to state the shapes of our tetrominos(O,Z,S,T,L,J,I)

```
// set the shape of the tetromino based on the given type
if (type == 'I') {
    // shape of the tetromino I in its initial orientation
    boolean [][] shape = {{false, true, false, false}, {false, true, false, false}, {false, true, false, false}
    shapeMatrix = shape;
    int [][] number = {{0,0,0,0},{0,0,0,0},{0,0,0,0},{0,0,0,0}};
    numberMatrix = number;

}
else if (type == 'S') {
    // shape of the tetromino S in its initial orientation
    boolean [][] shape = {{false, true, true}, {true, true, false}, {false, false, false}};
    shapeMatrix = shape;
    int [][] number = {{0,0,0},{0,0,0},{0,0,0}};
    numberMatrix = number;
}
else if (type == 'Z') {
    // shape of the tetromino Z in its initial orientation
    boolean [][] shape = {{true, true, false}, {false, true, true}, {false, false, false}};
    shapeMatrix = shape;
    int [][] number = {{0,0,0},{0,0,0},{0,0,0}};
    numberMatrix = number;
}
else if (type == 'O') {
    // shape of the tetromino O in its initial orientation
    boolean [][] shape = {{true, true}, {true, true}};
    shapeMatrix = shape;
    int [][] number = {{0,0},{0,0}};
    numberMatrix = number;
}
else if (type == 'T') {
```

Then we need to move the shape left and right and rotate it.

```
// Method for rotating tetromino clockwise by 90 degrees
public boolean rotate(Grid gameGrid) {
    // Check whether tetromino can rotate: all the blocks of the tetromino (including
    // the empty ones) must be inside the game grid and there must be no occupied
    // grid square within the blocks of the tetromino
    int n = shapeMatrix.length; // n = number of rows = number of columns for both shapeMatrix
    for (int row = 0; row < n; row++) {
        for (int col = 0; col < n; col++) {
            int positionX = coordinateMatrix[row][col].x;
            int positionY = coordinateMatrix[row][col].y;
            if (!gameGrid.isInside(positionY, positionX))
                return false;
            if (gameGrid.isOccupied(positionY, positionX))
                return false;
        }
    }
```

In the next step, if there is a filled line, we should delete it.

```
// Method used for checking whether the square with given indices is occupied or empty
public boolean isOccupied(int row, int col) {
    return colorMatrix[row][col] != emptySquare;
}
public boolean checkRowComplete(int row){
    for (int c = 0; c < colorMatrix[0].length; c++){
        if (!isOccupied(row, c)){
            return false;
        }
    }
    return true;
}
public boolean checkRowClear(int row){
    for (int c = 0; c < colorMatrix[0].length; c++){
        if (isOccupied(row, c)){
            return false;
        }
    }
    return true;
}
public void deleteRow(int rowToDelete, int firstClearRowAbove){
    // move rows (above rowToDelete) that contain occupied squares down by 1
    for (int row = rowToDelete; row < firstClearRowAbove - 1; row++){
        for (int col = 0; col < colorMatrix[0].length; col++){

            colorMatrix[row][col] = colorMatrix[row + 1][col];
            numberMatrix[row][col] = numberMatrix[row+1][col];
        }
    }
    // delete the row below firstClearRowAbove
    for (int col = 0; col < colorMatrix[0].length; col++){
```

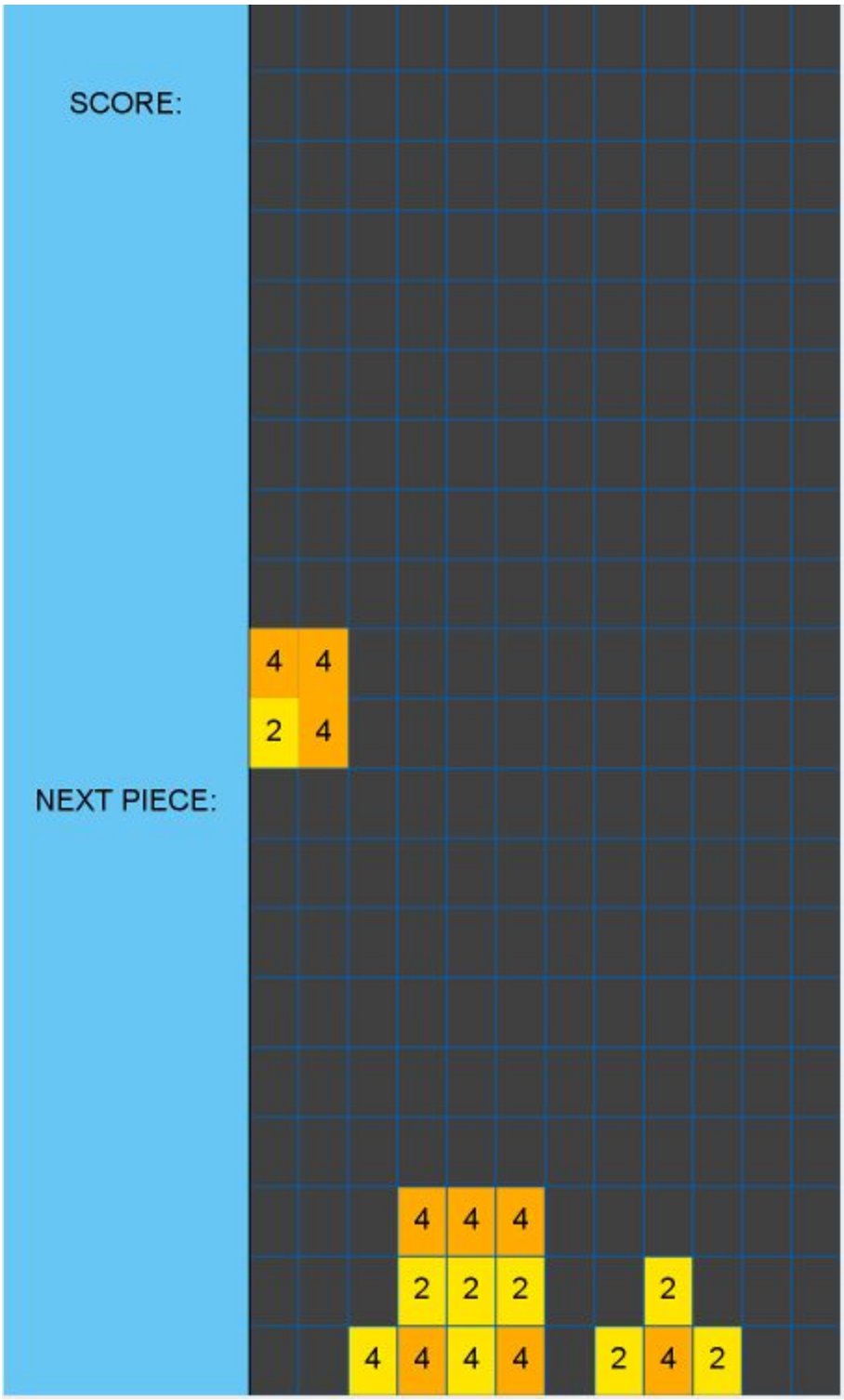
```
    // delete the row below firstClearRowAbove
    for (int col = 0; col < colorMatrix[0].length; col++){
        colorMatrix[firstClearRowAbove - 1][col] = emptySquare;
        numberMatrix[firstClearRowAbove - 1][col] = 0;
    }
```

So far, the basic parts of the tetris game have been made, so we need to combine it with the 2048 game and we need to open a new class in the tile name.

```
import java.awt.Color;
import java.awt.Point;
import java.util.Random;

public class tile {
    private int number; // number on the tile
    private Color backgroundColor;// color used for the tile
    private Point coordinateMatrix;
    private final static int[] nums= {2,4};

    tile(){
        Random r=new Random();
        int index= r.nextInt(2);
        number=nums[index];
        if(number==2) {
            backgroundColor=new Color(255,229,0);

        }
        else if(number==4) {


            backgroundColor=new Color(255,171,0);
        }
        else {
            backgroundColor = new Color(255,171-(2*number),0);
        }

    }
```

OUTPUT:

Before the shape falls



After the shape fell

List of achievements:

1. Using the stddraw library

   setPenColor(Color color)

   setXscale(double xmin, double xmax)

   setYscale(double ymin, double ymax)

   StdDraw.text(0.5, 0.5, "Hello, World");

2. Improving algorithmic thinking
3. The development of search ability
4. Strengthening the use of java oop

References:

- https://www.ssaurel.com/blog/learn-to-create-a-tetris-game-in-java-with-swing/
- http://zetcode.com/tutorials/javagamestutorial/tetris/
- https://introcs.cs.princeton.edu/java/stdlib/javadoc/StdDraw.html