Lab7（2018-11-11）

董依菡 15302010054@fudan.edu.cn

耿同欣 15302010048@fudan.edu.cn

张星宇 15307110273@fudan.edu.cn

任务

请大家在lab5的基础上完成以下功能

- 方块下落时接触到其他方块或容器底部时，堆积在容器里面，并进行下一个方块的下落。

```
| * *      |                    |    * *    |
|    * *   |      下落到底部后   |      * * |
|          |      ---------->   |           |
|          |      生成下一个块   | * *       |
|          |                    |    * *    |
-----------                    -----------
```

- 当容器中有一行的格子被小方块全部占领时，该行消除，该行上面的小方块自动下落一格

```
|          |                    |           |
|          |      第二行已占满   |           |
|    * *   |      ---------->   |           |
| * * * *  |      消除该行下落   |    * *    |
|    * *   |                    |    * *    |
-----------                    -----------
```

- 若方块溢出容器，游戏结束
- 只需要实现一种方块(Z)即可

此次lab建议使用**面向对象**的方法来重构之前的代码，并实现新的功能，建议大家阅读提示部分。

提示

参考的代码框架

- src/Main.java

```
/**
 * Any question, feel free to contact Xingyu Zhang.
 */
import java.util.Scanner;
public class Main {
```

```
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Game game = Game.getInstance();
        game.initial(10,10);
        game.nextBlock();
        while(true) {
            if (game.isOver()) {
                System.out.println("Game over!");
                break;
            }
            game.print();

            String command = scanner.next();
            game.executeCommand(command);

        }
    }
}
```

- src/ZBlock.java

```
/**
 * Any question, feel free to contact Xingyu Zhang.
 * class ZBlock, 3*3 cells.
 */
public class ZBlock {
    // use cells to describe the shape of ZBlock
    private boolean cells[][];
    // rows of cells
    private int rows;
    // columns of cells
    private int columns;

    // get and set methods
    // todo

    /**
     * Constructor of ZBlock, initialize cells with the shape of "Z".
     */
    public ZBlock() {
        // todo
    }

    /**
     * Get the next cells 90-degree rotated from current cells.
     * This method will not change the property "cells".
     * @return the rotated cells
     */
```

```
    public boolean[][] nextRotatedCells() {
        // todo
    }

    /**
     * rotate cells by 90 degrees.
     */
    public void rotate() {
        // todo
    }

}
```

- src/Game.java

```
/**
 * Any question, feel free to contact Xingyu Zhang.
 * Since we need only one instance of Game, we use "singleton-pattern"
 */
public class Game {

    // the cells are used to hold the blocks
    private boolean[][] cells;
    private int rows;
    private int columns;

    // current moving block
    private ZBlock currentBlock;
    // the global position of the cells[0][0] of currentBlock
    private int posRow;
    private int posColumn;

    // create an instance of Game
    private static Game game = new Game();


    /**
     * "private" makes sure that the instance cannot be created by invoking
constructor,
     * meaning that the "game" above became the single instance.
     */
    private Game() {
    }

    /**
     * get the single instance
     * @return the single instance of Game.
     */
    public static Game getInstance() {
```

```
        return game;
    }

    /**
     * initialize the cells with height and width, and assign all cells
with "false".
     * YOU NEED TO:
     * assign rows and columns with height and width
     * initialize the cells and assign them with "false".
     * @param height: the rows of cells
     * @param width: the columns of cells
     */
    public void initial(int height, int width) {
        // todo
    }


    /**
     * Get next block, invoked when the game started or the last block has
reached the bottom.
     * YOU NEED TO:
     * assign posRow and posColumn with 0 and columns / 2 - blockColumns /
2.
     * assign currentBlock with a new instance of ZBlock.
     */
    public void nextBlock() {
        // todo
    }

    /**
     * since posRow, posColumn, currentBlock will never collide with game
cells except when nextBlock appeared.
     * we can invoke "collide()" below to check whether the game is over.
     * @return: whether the game is over
     */
    public boolean isOver() {
        return collide(posRow, posColumn, currentBlock.getCells());
    }

    /***
     * print the game UI
     */
    public void print() {
        // todo
    }

    /***
     * execute next command, you may need to invoke collide() to check
collision before rotating or moving.
```

```
     * when the command is "s" and the block has reached to the bottom,
     * loadBlock(), eliminate(), nextBlock() will be invoked sequentially.
     * @param command: the next command {"w","s","a","d"}
     */
    public void executeCommand(String command) {
        // todo
    }

    /**
     * judge whether blockCells at position(nextRow, nextColumn) collide
with game cells.
     * @param nextPosRow: the next posRow after rotating or moving.
     * @param nextPosColumn: the next posColumn after rotating or moving.
     * @param nextBlockCells: the next blockCells after rotating or moving.
     * @return: true if collided, false otherwise.
     */
    private boolean collide(int nextPosRow, int nextPosColumn, boolean[][]
nextBlockCells){
        // todo
    }

    /**
     * assign "true" to the game cells occupied by currentBlock.
     */
    private void loadBlock() {
        // todo
    }

    /***
     * eliminate all the "full" rows. "full" means all true.
     */
    private void eliminate() {
        // todo
    }

    /***
     * Force to eliminate one row, and all the cells above it will drop by
one row.
     * @param row: the index of the eliminated row.
     */
    private void eliminateOneRow(int row) {
        // todo
    }
}
```

提交

提交地址:ftp://10.132.141.33/classes/18/181 程序设计A(戴开宇)/WORK_UPLOAD/lab7

提交内容:将java程序(.java文件)压缩成zip格式，zip压缩包的名称:学号_姓名_lab7.zip

截止止日日期:2018/11/18 23:59:59

祝大家节日快乐=。=