**Содержание**

Введение 3

Цель разработки 3

Описание алгоритма работы программы 3

Руководство для администратора 4

Руководство для пользователя 4

Исходный код программы 5

# Введение

Тетрис –  компьютерная игра, первоначально изобретённая и разработанная советским программистом Алексеем Пажитновым. Игра была выпущена 6 июня 1984 года — в это время Пажитнов работал в Вычислительном центре Академии наук СССР.

Правила игры – случайные фигурки тетрамино падают сверху в прямоугольный стакан шириной 10 и высотой 20 клеток. В полёте игрок может поворачивать фигурку на 90° и двигать её по горизонтали. Также можно «сбрасывать» фигурку, то есть ускорять её падение, когда уже решено, куда фигурка должна упасть. Фигурка летит до тех пор, пока не наткнётся на другую фигурку либо на дно стакана. Если при этом заполнился горизонтальный ряд из 10 клеток, он пропадает и всё, что выше него, опускается на одну клетку. Дополнительно показывается фигурка, которая будет следовать после текущей — это подсказка, которая позволяет игроку планировать действия. Темп игры постепенно увеличивается. Игра заканчивается, когда новая фигурка не может поместиться в стакан. Игрок получает очки за каждый заполненный ряд, поэтому его задача — заполнять ряды, не заполняя сам стакан (по вертикали) как можно дольше, чтобы таким образом получить как можно больше очков.

# Цель разработки

Создать аналог классической игры Тетрис на языке программирования Java.

# Описание алгоритма работы программы

Приложение представляет из себя простое окно без каких-либо кнопок и меню, управление производится четырьмя клавишами управления курсора: вверх, вниз, влево и вправо. Перед каждым перемещением фигуры проверяется свободное место около неё. При заполнении линии она уничтожается, если фигурам будет негде появляться, то игра окончится и на экране появится надпись «Game over». Более подробный алгоритм представлен в блок-схеме (Рис. 1).

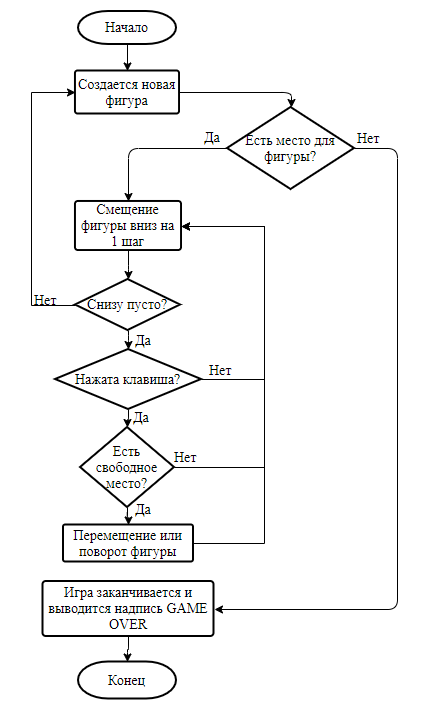


Рис. 1

# Руководство для администратора

1. Скачать приложение Tetris на компьютер.
2. Переместить в необходимую папку.

# Руководство для пользователя

1. Запустите приложение Tetris.jar.
2. Управляйте фигуркой клавишами перемещения курсора (стрелками).

# Исходный код программы

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import java.util.\*;

class Tetris extends JFrame {

final String TITLE\_OF\_PROGRAM = "Tetris";

final int BLOCK\_SIZE = 25;

final int ARC\_RADIUS = 6;

final int FIELD\_WIDTH = 10;

final int FIELD\_HEIGHT = 18;

final int START\_LOCATION = 180;

final int FIELD\_DX = 7;

final int FIELD\_DY = 26;

final int LEFT = 37;

final int UP = 38;

final int RIGHT = 39;

final int DOWN = 40;

final int SHOW\_DELAY = 200; // задержка

final int[][][] SHAPES = {

{{0,0,0,0}, {1,1,1,1}, {0,0,0,0}, {0,0,0,0}, {4, 0x00f0f0}}, // I

{{0,0,0,0}, {0,1,1,0}, {0,1,1,0}, {0,0,0,0}, {4, 0xf0f000}}, // O

{{1,0,0,0}, {1,1,1,0}, {0,0,0,0}, {0,0,0,0}, {3, 0x0000f0}}, // J

{{0,0,1,0}, {1,1,1,0}, {0,0,0,0}, {0,0,0,0}, {3, 0xf0a000}}, // L

{{0,1,1,0}, {1,1,0,0}, {0,0,0,0}, {0,0,0,0}, {3, 0x00f000}}, // S

{{1,1,1,0}, {0,1,0,0}, {0,0,0,0}, {0,0,0,0}, {3, 0xa000f0}}, // T

{{1,1,0,0}, {0,1,1,0}, {0,0,0,0}, {0,0,0,0}, {3, 0xf00000}} // Z

};

int [][] mine = new int[FIELD\_HEIGHT + 1][FIELD\_WIDTH];

JFrame frame;

Canvas canvasPanel = new Canvas();

Random random = new Random();

Figure figure = new Figure();

boolean gameOver = false;

final int[][] GAME\_OVER\_MSG = {

{0,1,1,0,0,0,1,1,0,0,0,1,0,1,0,0,0,1,1,0},

{1,0,0,0,0,1,0,0,1,0,1,0,1,0,1,0,1,0,0,1},

{1,0,1,1,0,1,1,1,1,0,1,0,1,0,1,0,1,1,1,1},

{1,0,0,1,0,1,0,0,1,0,1,0,1,0,1,0,1,0,0,0},

{0,1,1,0,0,1,0,0,1,0,1,0,1,0,1,0,0,1,1,0},

{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},

{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},

{0,1,1,0,0,1,0,0,1,0,0,1,1,0,0,1,1,1,0,0},

{1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0},

{1,0,0,1,0,1,0,1,0,0,1,1,1,1,0,1,1,1,0,0},

{1,0,0,1,0,1,1,0,0,0,1,0,0,0,0,1,0,0,1,0},

{0,1,1,0,0,1,0,0,0,0,0,1,1,0,0,1,0,0,1,0}};

public static void main(String[] args){

new Tetris().go();

}

void go(){

frame = new JFrame(TITLE\_OF\_PROGRAM); frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(FIELD\_WIDTH \* BLOCK\_SIZE + FIELD\_DX, FIELD\_HEIGHT \* BLOCK\_SIZE + FIELD\_DY);

frame.setLocation(START\_LOCATION, START\_LOCATION);

frame.setResizable(false);// не изменяемое окно по размерам

canvasPanel.setBackground(Color.black);

frame.addKeyListener(new KeyAdapter() {

@Override

public void keyPressed(KeyEvent e) {

if (!gameOver) {

if (e.getKeyCode() == DOWN) figure.drop();

if (e.getKeyCode() == UP) figure.rotate();

if (e.getKeyCode() == LEFT || e.getKeyCode() == RIGHT) figure.move(e.getKeyCode());

}

canvasPanel.repaint();

}

});

frame.getContentPane().add(BorderLayout.CENTER, canvasPanel);

frame.setVisible(true);

Arrays.fill(mine[FIELD\_HEIGHT], 1);

//main loop of game

while (!gameOver) {

try {

Thread.sleep(SHOW\_DELAY);

} catch (Exception e) { e.printStackTrace();}

canvasPanel.repaint();

if (figure.isTouchGround()){

figure.leaveOnTheGround();

checkFilling();

figure = new Figure();

gameOver = figure.isCrossGround();

} else {

figure.stepDown();

}

}

}

void checkFilling() {

int row = FIELD\_HEIGHT - 1;

int countFillRows = 0;

while (row > 0) {

int filled = 1;

for (int col = 0; col < FIELD\_WIDTH; col++)

filled \*= Integer.signum(mine[row][col]);

if (filled > 0) {

countFillRows++;

for (int i = row; i > 0; i--) System.arraycopy(mine[i-1], 0, mine[i], 0, FIELD\_WIDTH);

} else

row--;

}

}

class Figure {

private ArrayList<Block> figure = new ArrayList<Block>();

private int[][] shape = new int[4][4];

private int type, size, color;

private int x = 3, y = 0;

Figure(){

type = random.nextInt(SHAPES.length);

size = SHAPES[type][4][0];

color = SHAPES[type][4][1];

if (size == 4) y = -1;

for (int i = 0; i < size; i++)

System.arraycopy(SHAPES[type][i], 0, shape[i], 0, SHAPES[type][i].length);

createFromShape();

}

void createFromShape() {

for (int x = 0; x < size; x++)

for (int y = 0; y < size; y++)

if (shape[y][x] == 1) figure.add(new Block(x + this.x, y + this.y));

}

boolean isTouchGround() {

for (Block block : figure)

if (mine[block.getY() + 1][block.getX()] > 0)

return true;

return false;

}

boolean isCrossGround() {

for (Block block : figure) if (mine[block.getY()][block.getX()] > 0) return true;

return false;

}

void leaveOnTheGround() {

for (Block block : figure) mine[block.getY()][block.getX()] = color;

}

void stepDown() {

for (Block block : figure) block.setY(block.getY() + 1);

y++;

}

void drop() {

while (!isTouchGround()) stepDown();

}

boolean isTouchWall(int direction) {

for (Block block : figure) {

if (direction == LEFT && (block.getX() == 0 || mine[block.getY()][block.getX() - 1] > 0)) return true;

if (direction == RIGHT && (block.getX() == FIELD\_WIDTH - 1 || mine[block.getY()][block.getX() + 1] > 0)) return true;

}

return false;

}

void move(int direction) {

if (!isTouchWall(direction)) {

int dx = direction - 38;

for (Block block : figure) block.setX(block.getX() + dx);

x += dx;

}

}

boolean isWrongPosition() {

for (int x = 0; x < size; x++)

for (int y = 0; y < size; y++)

if (shape[y][x] == 1) {

if (y + this.y < 0) return true;

if (x + this.x < 0 || x + this.x > FIELD\_WIDTH - 1) return true;

if (mine[y + this.y][x + this.x] > 0) return true;

}

return false;

}

void rotate() {

for (int i = 0; i < size/2; i++)

for (int j = i; j < size-1-i; j++) {

int tmp = shape[size-1-j][i];

shape[size-1-j][i] = shape[size-1-i][size-1-j];

shape[size-1-i][size-1-j] = shape[j][size-1-i];

shape[j][size-1-i] = shape[i][j];

shape[i][j] = tmp;

}

if (!isWrongPosition()) {

figure.clear();

createFromShape();

}

}

void paint(Graphics g) {

for (Block block : figure) block.paint(g, color);

}

}

class Block {

private int x,y;

public Block(int x, int y) {

setX(x);

setY(y);

}

void setX(int x) { this.x = x; }

void setY(int y) { this.y = y; }

int getX() { return x; }

int getY() { return y; }

void paint(Graphics g, int color) {

g.setColor(new Color(color));

g.drawRoundRect(x\*BLOCK\_SIZE+1, y\*BLOCK\_SIZE+1, BLOCK\_SIZE-2, BLOCK\_SIZE-2, ARC\_RADIUS, ARC\_RADIUS);

}

}

public class Canvas extends JPanel {

@Override

public void paint(Graphics g) {

super.paint(g);

for (int x = 0; x < FIELD\_WIDTH; x++)

for (int y = 0; y < FIELD\_HEIGHT; y++)

if (mine[y][x] > 0) {

g.setColor(new Color(mine[y][x]));

g.fill3DRect(x\*BLOCK\_SIZE + 1, y\*BLOCK\_SIZE + 1, BLOCK\_SIZE - 1, BLOCK\_SIZE - 1, true);

}

if (gameOver) {

g.setColor(Color.white);

for (int y = 0; y < GAME\_OVER\_MSG.length; y++)

for (int x = 0; x < GAME\_OVER\_MSG[y].length; x++)

if (GAME\_OVER\_MSG[y][x] == 1) g.fill3DRect(x\*11+18, y\*11+160, 10, 10, true);

} else

figure.paint(g);

}

}

}