МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ КОМП’ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра економічної кібернетики

та інформатики

Комплексне практичне індивідуальне завдання

з дисципліни

«Конструювання програмного забезпечення»

на тему: «Реалізація гри “Тетріс” на Java»

Виконав:

студент групи ПЗС-32

Ревук Владислав Русланович

Перевірив:

Сирник О. Й.

Тернопіль 2015

Зміст

Вступ

1.Аналіз предметної області та постановка задачі

1.1 Аналіз предметної області

1.2 Постановка задачі

2.Проект програми

2.1 Ієрархія об’єктів

2.2 Ієрархія спадкування

2.3 Діаграма станів гри

2.4 Алгоритмізація

2.5 Кодування і вибір інструментальних засобів

ВИСНОВОК

Література

Додатки

Додаток А Опис класів

Додаток Б Лістинг частини коду програми

Додаток В Результат роботи програми

Вступ

В даній роботі реалізується гра «Тетріс». Зміст гри полягає у тому, що випадкові фігурки падають зверху в прямокутний стакан висотою 10 і шириною 20 кліток. У польоті гравець може повертати фігурку і рухати її по горизонталі, але не уповільнювати політ. Фігурка летить, поки не натикнеться на іншу фігурку або на підлогу стакана. Якщо при цьому заповнився горизонтальний ряд з 10 кліток, він пропадає і все, що вище за нього, опускається на 1 клітку. Темп гри поступово збільшується.

При реалізації програми буде використовуватись об’єктно-орієнтований підхід, важливим наслідком чого є можливість формування такої структури програми, коли додавання нових компонентів при її подальшому розвитку не буде впливати на існуючи компоненти, або такий вплив буде зведено до мінімуму.

Для кодування гри буде обрана мова Java, що сприяє написанню надійного програмного продукту.

1.Аналіз предметної області та постановка задачі

1.1 Аналіз предметної області

Метою даної роботи є розробка гри „Тетріс”. Вона відноситься до логічних ігор, які допомагають розвинути реакцію, тренують логіку. При тривалій роботі за комп'ютером виникає бажання трохи відпочити. Для цієї мети не підходять ігри типу стратегій,бо для них потрібно багато часу, а для 5-10 хвилинного відпочинку цілком підійде саме «Тетріс». У стандартній поставці операційних систем даної гри немає, тому виникла необхідність у її створенні.

До ігор такого типу пред‘являються такі вимоги: використання простих засобів управління; зручний графічний інтерфейс; поступове ускладнення гри при наборі певної кількості очок. Під час виконання програма повинна виконуватися коректно і не приводити до збоїв.

1.2 Постановка задачі

Зміст гри полягає у прагненні щільно зайняти ігрове поле падаючими геометричними фігурами, змінюючи їх орієнтацію у просторі, домагаючись відсутності пропусків в кожному рядку.

Фігурине повинні виходити за кордони ігрового поля, вони мають переміщуватись та перевертатись. Для надання більшої привабливості зовнішньому вигляду ігрового поля при написанні гри потрібновикористовувати яскраву графіку. Інтерфейс програми, що розроблюється повинний бути зручний і зрозумілий користувачу.

Програма повинна реагувати на натискання клавіш клавіатури, виводячи зображення на екран.

При реалізації програми було вирішено використовуватиоб'єктно-орієнтований підхід, що дозволить уникнути проблем проектування, характерних для процедурного підходу. Важливим наслідком цього є можливість формування такої структури програми, коли додавання нових компонентів при її подальшому розвитку не буде впливати на існуючі компоненти, або такий вплив буде зведено до мінімуму. Відповідно до 1.1 і 1.2 розробимо наступний проект програми.

2.Проект програми

2.1 Ієрархія об’єктів

Аналізуючи потоки вхідних і вихідних даних, розробляємо загальний інтерфейс програми, уточнюємо вимоги до функцій, що виконуються програмою визначаємо їх структуру.

На етапі ескізного проектування були виділені такі об’єкти, що входять до складу програми: власне гра «Тетріс»; ігрове вікно, яке включає ігрове поле, поле кнопок і ігрове меню. Графічно це відображається в ієрархії об’єктів на рисунку 1.

Гра «Тетріс»

Ігрове вікно

Поле індікації

Ігрове меню

Ігрове поле

Рівень

Рахунок

Поточна фігура

Наступна фігура

Поле кнопок

Продов-жити гру

Призупи-нитигру

Почати гру спочатку

Зупинити гру

Вийти з гри

Рисунок 1 – Ієрархія об’єктів

2.2 Ієрархія спадкування

З урахуванням прийнятої методології і ієрархії класів нам необхідно ієрархію спадкування, яка зображена на рисунку 2.

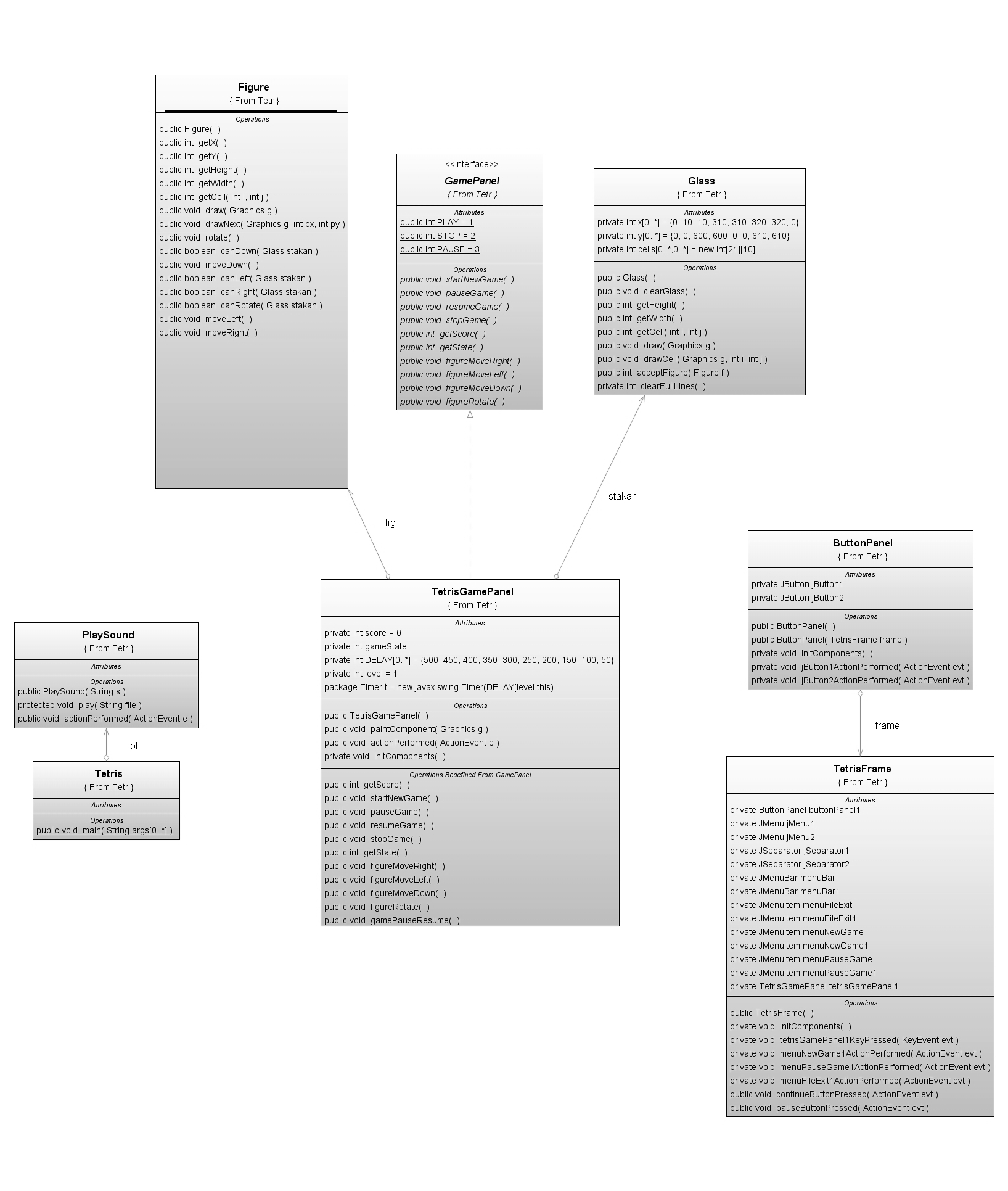


Рисунок 2 – Ієрархія спадкування

2.3 Діаграма станів гри

Діаграма станів гри зображена на рисунку 3.

Запуск Вихід

Натиснута кнопка «Pause» Обраний пункт меню «StartNewGame»

Обраний пункт меню «StopGame»

Натиснута кнопка «Resume»

Фігура досягла верху стакана

Рисунок 3 - Діаграма станів

2.4 Алгоритмізація

Одною з найважливіших функцій у даній роботі є очищення заповнених ліній. ЇЇ блок-схема показана на рисунку 4.

Початок

i=1; i<20; i++

j=0; j<10; j++

cells[i][j]==0

continue lineLoop

linesCount++

j=1; j>=10; j--

cells[j]=cells[j-1]

cells[0] = new int[10]

j=0; j<10; j++

cells[0][j]=0

Кінець

Рисунок 4 – Блок-схема методу private int clearFullLines()

2.5 Кодування і вибір інструментальних засобів

Враховуючи поставлену задачу, ієрархію класів, об’єктів, було зроблено висновок, що для даної роботи найкраще підійде мова програмування Java,так як використовуючи її маємо змогу використовувати величезний набір класів, які входять в стандартний пакет Java. Завдяки чому, наша робота включатиме розробку взаємодії між готовими класами. Також Java дозволяє досить легко створювати програми з графічним інтерфейсом користувача.

Висновок

У даній роботі був розроблений програмний продукт – логічна гра Тетріс.

Проведені усі необхідні роботи попроектуванню архітектури гри, проаналізовані вимоги до неї, приводиться опис реалізації, кодування, тестування програми.

При написанні програми була використана об’єктно-орієнтована мова Java.

Література

1. Хабибуллин И.Ш. Самоучитель Java 2. СПб.: БХВ-Петербург, 2007. – 720 с.: ил.
2. Эккель Б. Философия Java. Библиотека программиста. СПб.: Питер, 2001
3. БеркунськийЄ.Ю. Об’єктно-орієнтоване програмування мовою Java: Методичні вказівки для студентів напрямку “ Комп’ютерні науки”. – Миколаїв: НУК, 2006. – 52 с.

Додатки

В даному розділі буде детально розглянуто кожен з класів даної роботи, його поля і методи

Додаток А Опис класів

interface GamePanel – інтерфейс гральної панелі

|  |  |
| --- | --- |
| Поле | Опис поля |
| final static int PLAY = 1 | Стан «гра в процесі» |
| final static int STOP = 2 | Стан «гра зупинена» |
| final static int PAUSE =3 | Стан «гра призупинена» |

|  |  |
| --- | --- |
| Метод | Опис методу |
| public void startNewGame() | Почати нову гру |
| public void pauseGame() | Призупинити гру |
| public void resumeGame() | Продовжити гру |
| public void stopGame() | Зупинити гру |
| public int getScore() | Одержати рахунок |
| public int getState() | Одержати стан гри |
| public void figureMoveRight() | Рухати фігуру вправо |
| public void figureMoveLeft() | Рухати фігуру вліво |
| public void figureMoveDown() | Рухати фігуру вниз |
| public void figureRotate() | Обертати фігуру |
| public void gamePauseResume() | Змінити стан гри PAUSE на PLAY і навпаки |

public class Tetris – основний клас

|  |  |
| --- | --- |
| Метод | Опис методу |
| public static void main(String[] args) | Запуск програми |

public class TetrisFrame – вікно програми

|  |  |
| --- | --- |
| Метод | Опис методу |
| public TetrisFrame() | Конструктор класу TetrisFrame |
| private void tetrisGamePanel1KeyPressed (java.awt.event.KeyEvent evt) | Обробка натискання клавіш управління |
| private void menuNewGame1ActionPerformed (java.awt.event.ActionEvent evt) | Обробка вибору пункту меню «Start New Game» |
| private void menuPauseGame1ActionPerformed (java.awt.event.ActionEvent evt) | Обробка вибору пункту меню «Stop Game» |
| private void menuFileExit1ActionPerformed (java.awt.event.ActionEvent evt) | Обробка вибору пункту меню «Exit» |
| public void continueButtonPressed (ActionEvent evt) | Обробка натискання кнопки «Resume» |
| public void pauseButtonPressed  (ActionEvent evt) | Обробка натискання кнопки «Pause» |

public class PlaySound – програвання музики

|  |  |
| --- | --- |
| Метод | Опис методу |
| public PlaySound(String s) | Конструктор класу PlaySound |
| protected void play(String file) | Програвання музики |

public class Figure – клас фігури

|  |  |
| --- | --- |
| Поле | Опис поля |
| final static byte[][][] PATTERN | Шаблони фігур |
| final static Color[] COLORS | Шаблони кольорів фігур |
| private int type | Тип фігури |
| private int[][] pat | Шаблон фігури |
| private int x | Координата х |
| private int y | Координата у |
| private int rotation | Вугол |
| private static int nextType | Наступний тип фігури |
| private static int nextRot | Наступний вугол |

|  |  |
| --- | --- |
| Метод | Опис методу |
| public Figure() | Конструктор класу Figure |
| public int getX() | Одержати х |
| public int getY() | Одержати у |
| public int getHeight() | Одержати висоту |
| public int getWidth() | Одержати ширина |
| public int getCell(int i, int j) | Одержати вміст фігури |
| public void draw(Graphics g) | Малювати фігуру |
| public void drawNext(Graphics g, int px, int py) | Малювати наступну фігуру |
| public void rotate() | Повернути фігуру |
| public boolean canDown | Чи можна рухати фігуру вниз? |
| public void moveDown() | Рухати фігуру вниз |
| public boolean canLeft(Glass stakan) | Чи можна рухати фігуру вліво? |
| public boolean canRight(Glass stakan) | Чи можна рухати фігуру управо? |
| public boolean canRotate(Glass stakan) | Чи можна повертати фігуру? |
| public void moveLeft() | Рухати фігуру вліво |
| public void moveRight() | Рухати фігуру управо |

public class Glass – клас стакан

|  |  |
| --- | --- |
| Поле | Опис поля |
| private int[] x | Координата х стакана |
| private int[] y | Координата у стакана |
| private int[][] cells | Вміст стакана |

|  |  |
| --- | --- |
| Метод | Опис методу |
| public Glass() | Конструктор класу Glass |
| public void clearGlass() | Очищення стакана |
| public int getHeight() | Одержати висоту стакана |
| public int getWidth() | Одержати ширину стакана |
| public int getCell(int i, int j) | Одержати вміст стакана |
| public void draw(Graphics g) | Малювання стакана |
| public void drawCell(Graphics g, int i, int j) | Малювання осередку стакана |
| public int acceptFigure(Figure f) | Прийняття стаканом фігури, що впала |
| private int clearFullLines() | Очищення повністю заповнених ліній |

public class ButtonPanel – панель кнопок

|  |  |
| --- | --- |
| Метод | Опис методу |
| public ButtonPanel(TetrisFrame frame) | Конструктор класу ButtonPanel |
| private void jButton1ActionPerformed (java.awt.event.ActionEvent evt) | Обробка натискання кнопки «Resume» |
| private void jButton2ActionPerformed (java.awt.event.ActionEvent evt) | Обробка натискання кнопки «Pause» |

public class TetrisGamePanel – ігрова панель

|  |  |
| --- | --- |
| Поле | Опис поля |
| private Glass stakan | Стакан тетріса |
| private Figure fig | Поточна фігура |
| private int score | Рахунок гри |
| private int gameState | Стан гри |
| private int[] DELAY | Паузи для падіння фігури по рівнях |
| private int level | Рівень гри |

|  |  |
| --- | --- |
| Метод | Опис методу |
| public TetrisGamePanel() | Конструктор класу TetrisGamePanel |
| public void paintComponent(Graphics g) | Малювання стакана, фігури, рахунка і рівня гри |
| public int getScore() | Поточний рахунок гри |
| public void startNewGame() | Почати нову гру |
| public void pauseGame() | Призупинення гри |
| public void resumeGame() | Продовження гри після паузи |
| public void stopGame() | Зупинити гру |
| public int getState() | Отримати стан гри |
| public void figureMoveRight() | Рухати фігуру вправо |
| public void figureMoveLeft() | Рухати фігуру вліво |
| public void figureMoveDown() | Рухати фігуру вниз |
| public void figureRotate() | Обертати фігуру |
| public void gamePauseResume() | Змінити стан гри PAUSE на PLAY і навпаки |
| public void actionPerformed(ActionEvent e) | Обробка дій, подій таймера |

Додаток Б Лістинг програми

ФайлGamePanel.java

package Tetr;

import java.awt.event.\*;

interface GamePanel extends ActionListener{

final static int PLAY = 1;

final static int STOP = 2;

final static int PAUSE =3;

public void startNewGame();

public void pauseGame();

public void resumeGame();

public void stopGame();

public int getScore();

public int getState();

public void figureMoveRight();

public void figureMoveLeft();

public void figureMoveDown();

public void figureRotate();

public void gamePauseResume();

}

ФайлTetris.java

package Tetr;

public class Tetris {

public static void main(String[] args) throws InterruptedException {

TetrisFrame.setDefaultLookAndFeelDecorated(true);

TetrisFrame frame = new TetrisFrame();

frame.setVisible(true);

String s = "D:\\FirstProb2\\src\\Tetr\\tetris.mid";

if (args.length>0)

s = args[0];

new PlaySound(s);

}

}

ФайлButtonPanel.java

package Tetr;

public class ButtonPanel extends javax.swing.JPanel {

private TetrisFrame frame;

public ButtonPanel(TetrisFrame frame) {

this.frame = frame;

initComponents();

}

private void initComponents() {

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

setFocusable(false);

jButton1.setText("Resume");

jButton1.setFocusable(false);

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

add(jButton1);

jButton2.setText("Pause");

jButton2.setFocusable(false);

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

add(jButton2);

}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

frame.continueButtonPressed(evt);

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

frame.pauseButtonPressed(evt);

}

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

}

Додаток В Результат роботи програми

Результат роботи програми зображено на рисунку 5

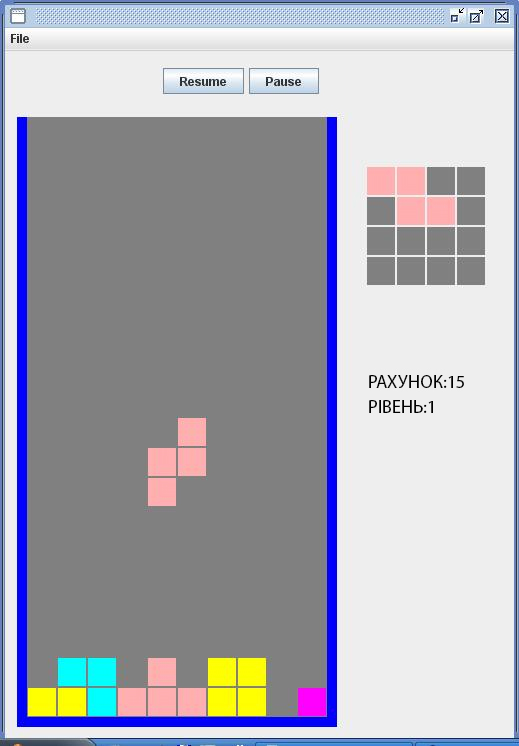


Рисунок 5 – Результат роботи програми