Mini Project Game OOP

Tetris 2.0

จัดทำโดย

นายพงษ์พัฒน์ บางข่า 6604062630358

เสนอ

ผู้ช่วยศาสตราจารย์ สถิตย์ ประสมพันธ์

รายงานนี้เป็นส่วนหนึ่งของรายวิชา **Object-Oriented Programming**

ภาควิชาวิทยาการคอมพิวเตอร์และสารสนเทศ คณะวิทยาศาสตร์ประยุกต์

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

ภาคเรียนที่ 1 ปีการศึกษา 2567

คำนำ

รายงานฉบับนี้ จัดทำขึ้นเพื่อเป็นส่วนหนึ่งของรายวิชา รายวิชา 040613204 การโปรแกรมเชิงวัตถุ (Object-Oriented Programming) สำหรับศึกษาในหัวข้อ Mini project โดยเนื้อหาของรายงานเกี่ยวข้องกับ การสร้างเกมโดยใช้งานภาษา Java โดยจุดมุ่งหมายของรายงายนี้ต้องการให้ผู้มาอ่านสามารถต่อยอดองค์ความรู้ หรือเป็นแรงบันดาลใจต่างๆ

สุดท้ายนี้ผู้เขียนหวังเป็นอย่างยิ่งว่า เอกสารรายงานนี้จะเป็นประโยชน์ แก่ผู้ที่ได้อ่านเพื่อต่อยอดในในอนาคต ไม่มากก็น้อย

นายพงษ์พัฒน์ บางข่า

ผู้เรียบเรียง

สารบัญ

หัวข้อที่ 1

บทนำ

1.1 ที่มาและความสำคัญของโปรเจ็ค

โปรเจคเกม Tetris 2.0 เป็นการต่อยอดจากเกมที่เป็นตำนานอย่าง Tetris ที่เปิดตัว วันที่ 6 มิถุนายน ค.ศ.1984/พ.ศ.2527 สำหรับเล่นในเครื่อง Electronika 60 แต่ใน Tetris 2.0 นี้จะเป็นการเพิ่มความท้าทาย และความน่าสนใจในการเล่นเกมตัวต่อนี้ โดย ผมจะทำการเพิ่มลูกเล่นต่างๆ ให้กับเกม เช่น สามารถเล่นกันได้ 2 คน , มีไอเทมให้สามารถใช้งานได้ เป็นต้น

1.2 ประเภทของโครงการ

เกมต่อตัวต่อ Tetris แบบ ผู้เล่น 2 คน

1.3 ประโยชน์

1. ฝึกไหวพริบในการตัดสินใจ
2. ฝึกความเร็วในการตอบสนอง Reaction Time
3. ฝึกความอดทน
4. เพื่อความสนุกสนาน

1.4 ขอบเขตของโครงการ

ลำดับ	รายการ	14-20	21-27	28-30
1	หารูปตัวละครและทำกราฟิกต่างๆ			
2	ศึกษาเอกสารและข้อมูลที่เกี่ยวข้อง			
3	ลงมือเขียนโปรแกรม			
4	จัดทำเอกสาร			
5	ตรวจสอบและแก้ไขข้อผิดพลาด			

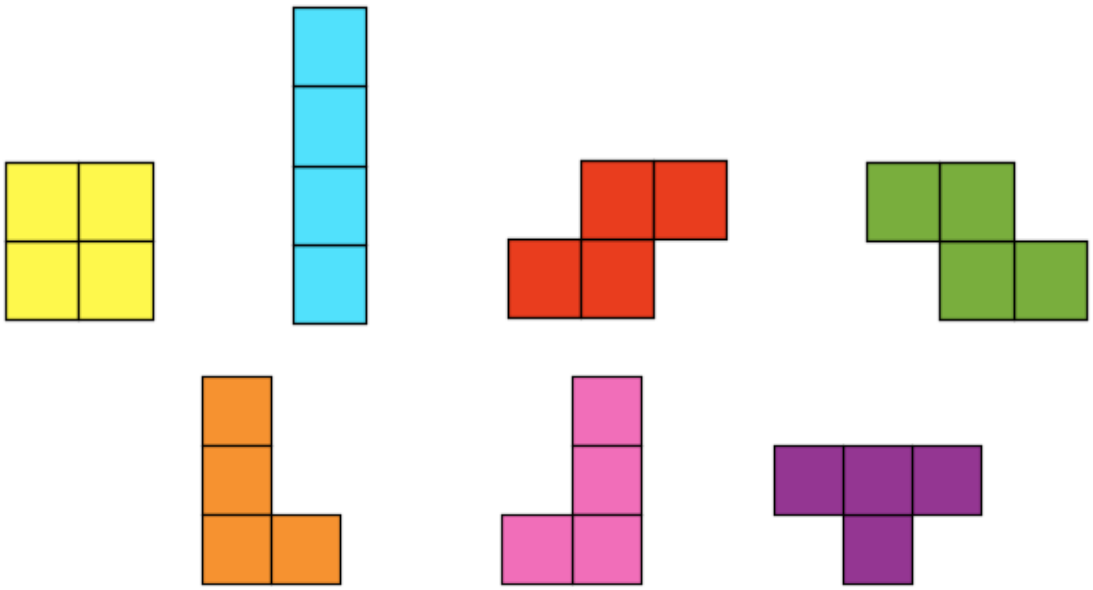
หัวข้อที่ 2

ส่วนการพัฒนา

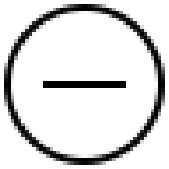
2.1 การเล่น

P1	P2	
A , D	←, →	เพื่อขยับตัวต่อไปซ้าย หรือ ขาว
W	↑	เพื่อหมุนตัวต่อ
S	↓	เพื่อให้ตัวต่อ ลงมาเร็ว

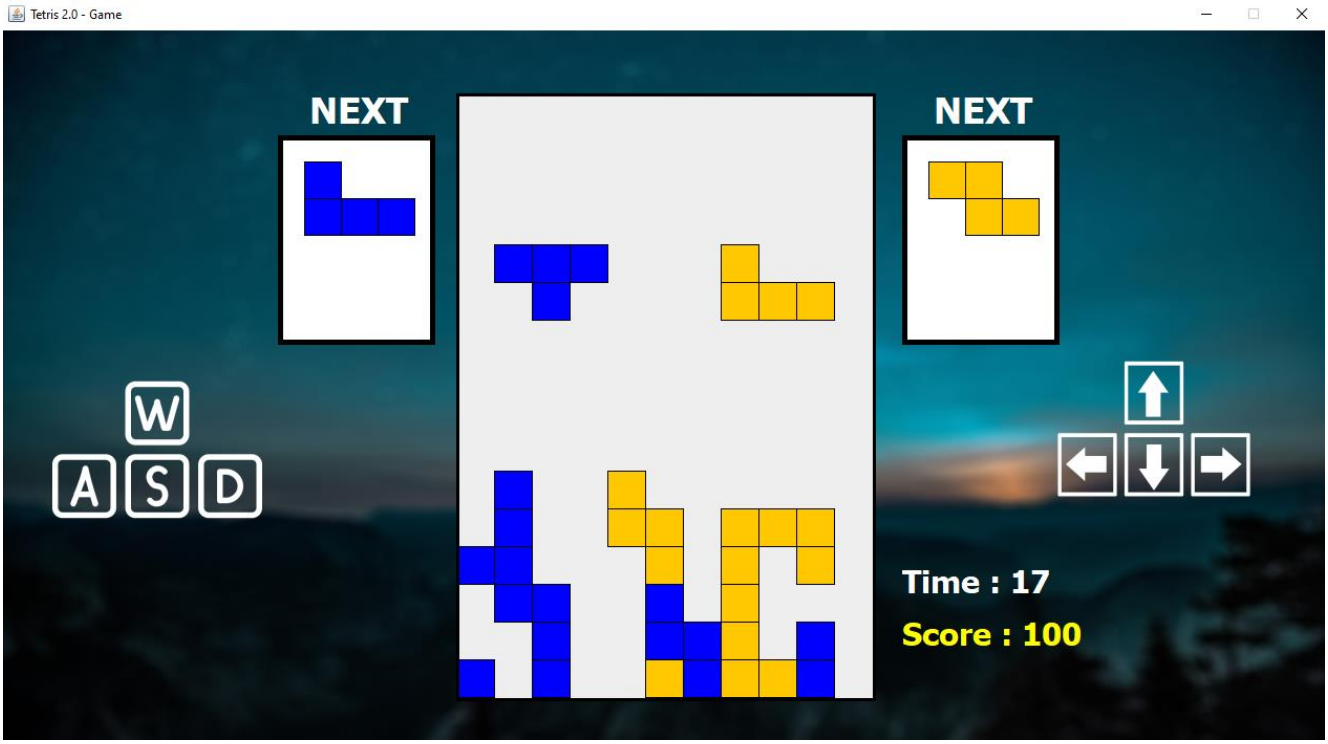
- Box / ตัวต่อ

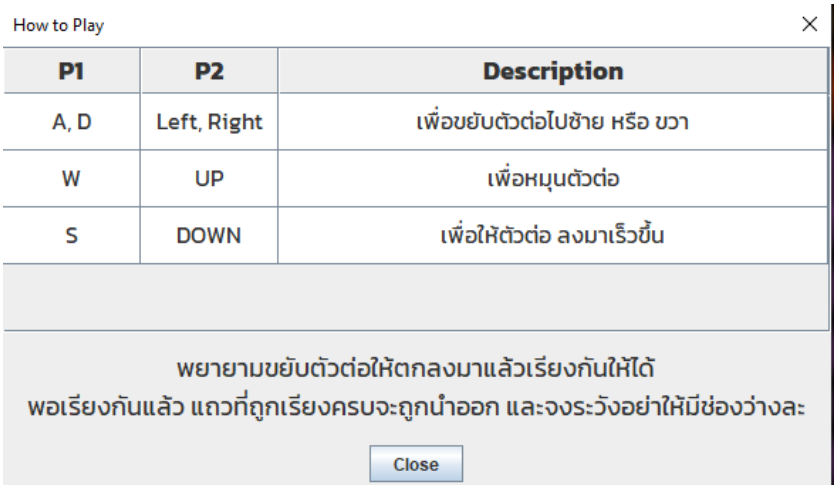


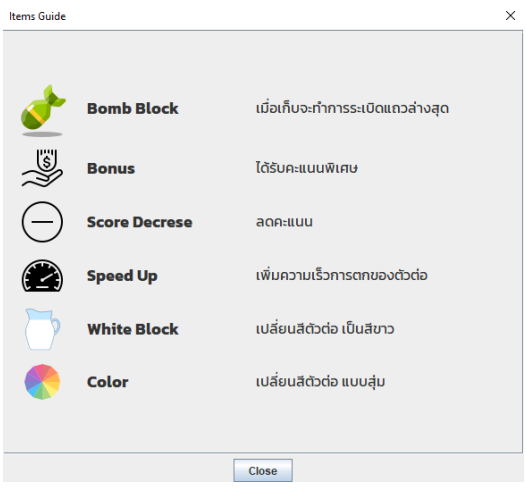
- Item / ของใช้

Bomb Block	เมื่อเก็บจะทำการระเบิดแถวล่างสุด	
Bonus	ได้รับคะแนนพิเศษ	
Score Decrese	ลดคะแนน	
Speed Up	เพิ่มความเร็วการตกของตัวต่อ	
White Block	เปลี่ยนสีตัวต่อ เป็นสีขาว	
Color	เปลี่ยนสีตัวต่อ แบบสุ่ม	

- Scene / ฉาก





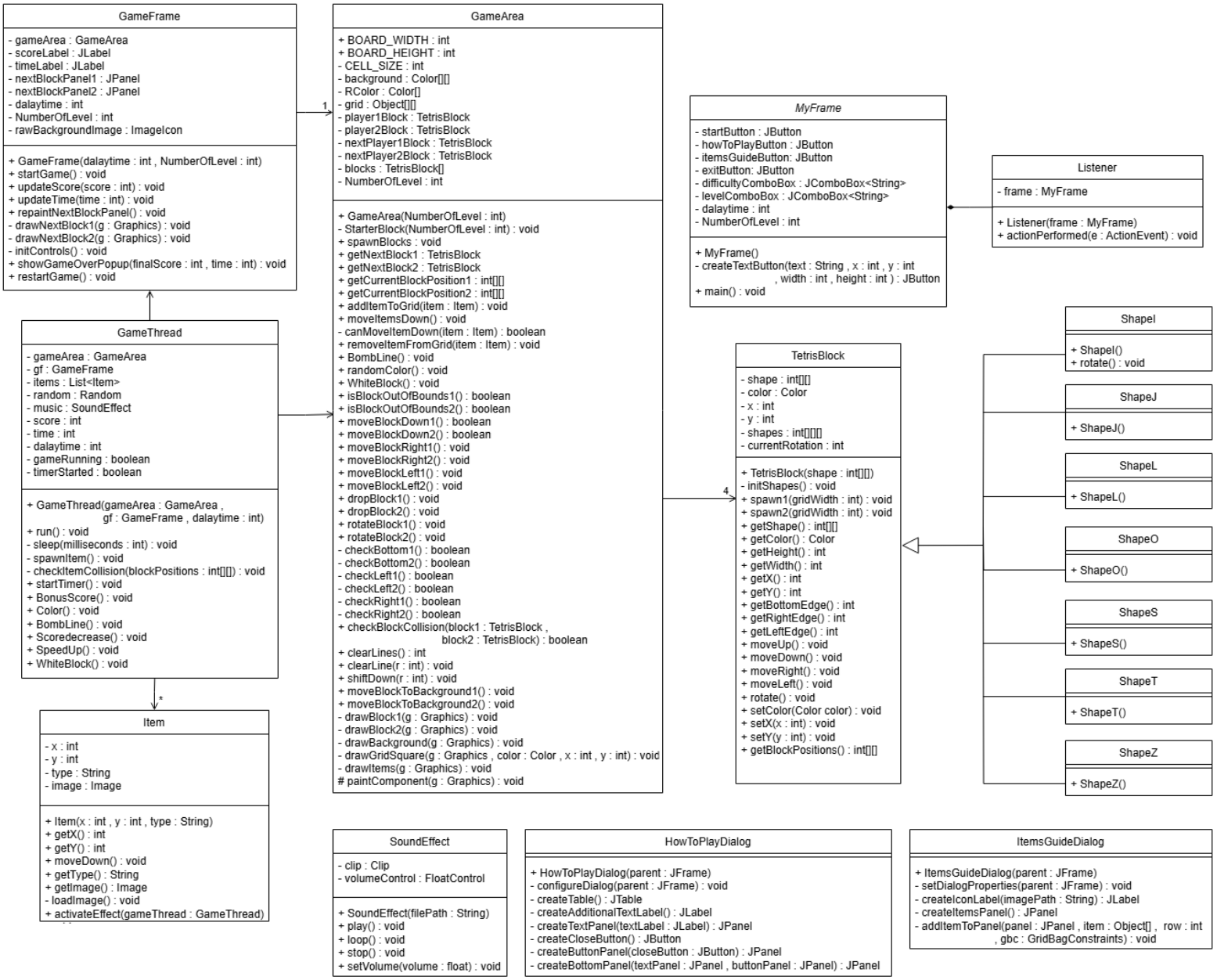


2.2 คลาสไดอะแกรม

Project นี้จะมีคลาสหลักอยู่ 9 คลาสคือ

1. Class MyFrame – จะจัดการหน้าต่างเมนูเริ่มเกม
2. Class GameArea – จะควรคุมการเล่นเกม
3. Class GameFrame – จะจัดการหน้าต่างแสดงผลของเกม
4. Class GameThread – จะควบคุมเวลาของเกม
5. Class HowToPlayDialog – จะแสดงหน้าต่างอธิบายการเล่นเกม
6. Class ItemsGuideDialog – จะแสดงหน้าต่างอธิบาย Item ในเกม

7. Class Item – จะเก็บข้อมูล Item ในเกม
8. Class TetrisBlock – จะจัดการการควบคุม Block ของผู้เล่น
9. Class SoundEffect – จะจัดการเรื่องเสียงในเกม



2.3 รูปแบบการพัฒนา

- ภาษา : Java

- GUI : javax.swing , java.awt

- ระบบเสียง : javax.sound.sampled.*;

- โปรแกรมตัดรูป: Photoshop

2.4 แนวคิดการเขียนโปรแกรมเชิงวัตถุ

- Constructor

```
MyFrame() {
        startButton = createTextButton("START", 0, 250, 400, 100);
        startButton.setFont(new Font("Tahoma", Font.BOLD, 100));
        howToPlayButton = createTextButton("How to Play", 0, 350, 450, 80);
        itemsGuideButton = createTextButton("Items Guide", 0, 425, 450, 80);
        exitButton = createTextButton("Exit Game", 0, 550, 300, 80);
        exitButton.setFont(new Font("Tahoma", Font.PLAIN, 40));
        exitButton.setForeground(new Color(192, 192, 192));

        String[] difficultyLevels = { "Easy", "Normal", "Hard" };
        difficultyComboBox = new JComboBox<>(difficultyLevels);
        difficultyComboBox.setBounds(1050, 50, 150, 50);
        difficultyComboBox.setFont(new Font("Tahoma", Font.BOLD, 30));
        difficultyComboBox.setOpaque(true);
        difficultyComboBox.setFocusable(false);
        difficultyComboBox.setBorder(null);

        String[] gameLevels = { "Level 1", "Level 2", "Level 3", "Level 4", "Level 5", "Level 6" }
;       levelComboBox = new JComboBox<>(gameLevels);
        levelComboBox.setBounds(1050, 120, 150, 50);
        levelComboBox.setFont(new Font("Tahoma", Font.BOLD, 30));
        levelComboBox.setOpaque(true);
        levelComboBox.setFocusable(false);
        levelComboBox.setBorder(null);
```

```
Listener listener = new Listener(this);
        startButton.addActionListener(listener);
        howToPlayButton.addActionListener(listener);
        itemsGuideButton.addActionListener(listener);
        exitButton.addActionListener(listener);
        difficultyComboBox.addActionListener(listener);
        levelComboBox.addActionListener(listener);

        ImageIcon rawLogoImage = new ImageIcon(getClass().getResource("logoW.png"));
        Image scaledLogoImage = rawLogoImage.getImage().getScaledInstance(600, 200, Image.SCALE_SMOOTH);

        ImageIcon logoImage = new ImageIcon(scaledLogoImage);
        JLabel logoLabel = new JLabel(logoImage);
        logoLabel.setBounds(20, -20, 600, 200);

        ImageIcon rawBackgroundImage = new ImageIcon(getClass().getResource("BG.jpg"));
        Image scaledBackgroundImage = rawBackgroundImage.getImage().getScaledInstance(1280, 720, Image.
SCALE_SMOOTH);
        ImageIcon backgroundImage = new ImageIcon(scaledBackgroundImage);
        JLabel background = new JLabel(backgroundImage);
        background.setBounds(0, 0, 1280, 720);

        add(logoLabel);
        add(startButton);
        add(howToPlayButton);
        add(itemsGuideButton);
        add(exitButton);
        add(difficultyComboBox);
        add(levelComboBox);
        add(background);
    }
```

Constructor ของ Class MyFrame จะเป็นการจัดการ Layout ของ Components ต่างๆ

```
public GameFrame(int dalaytime ,int NumberOfLevel) {
        this.dalaytime = dalaytime;
        this.NumberOfLevel = NumberOfLevel;
        switch(this.NumberOfLevel){
            case 0 : rawBackgroundImage = new ImageIcon(getClass().getResource("BG2.jpg")); break;
            case 1 : rawBackgroundImage = new ImageIcon(getClass().getResource("BG3.jpg")); break;
            case 2 : rawBackgroundImage = new ImageIcon(getClass().getResource("BG4.jpg")); break;
            case 3 : rawBackgroundImage = new ImageIcon(getClass().getResource("BG5.jpg")); break;
            case 4 : rawBackgroundImage = new ImageIcon(getClass().getResource("BG6.jpg")); break;
            case 5 : rawBackgroundImage = new ImageIcon(getClass().getResource("BG7.jpg")); break;
        }
        Image scaledBackgroundImage = rawBackgroundImage.getImage().getScaledInstance(1280, 720, Image.
SCALE_SMOOTH);
        ImageIcon backgroundImage = new ImageIcon(scaledBackgroundImage);
        JLabel background = new JLabel(backgroundImage);
        background.setBounds(0, 0, 1280, 720);

        gameArea = new GameArea(NumberOfLevel);
        gameArea.setBorder(BorderFactory.createLineBorder(Color.black,3));
        gameArea.setBounds(435, 60, 400, 580);

        JLabel nextBlockLabel1 = new JLabel("NEXT");
        nextBlockLabel1.setBounds(295, 50, 100, 50);
        nextBlockLabel1.setForeground(Color.WHITE);
        nextBlockLabel1.setFont(new Font("Tahoma", Font.BOLD, 35));

        nextBlockPanel1 = new JPanel() {
            @Override
            protected void paintComponent(Graphics g) {
                super.paintComponent(g);
                drawNextBlock1(g);
            }
        };
        nextBlockPanel1.setBounds(265, 100, 150, 200);
        nextBlockPanel1.setBorder(BorderFactory.createLineBorder(Color.black, 5));
        nextBlockPanel1.setBackground(Color.WHITE);
```

```
JLabel nextBlockLabel2 = new JLabel("NEXT");
        nextBlockLabel2.setBounds(860, 50, 100, 50);
        nextBlockLabel2.setForeground(Color.WHITE);
        nextBlockLabel2.setFont(new Font("Tahoma", Font.BOLD, 35));

        nextBlockPanel2 = new JPanel() {
            @Override
            protected void paintComponent(Graphics g) {
                super.paintComponent(g);
                drawNextBlock2(g);
            }
        };
        nextBlockPanel2.setBounds(860, 100, 150, 200);
        nextBlockPanel2.setBorder(BlockBorderFactory.createLineBorder(Color.black, 5));
        nextBlockPanel2.setBackground(Color.WHITE);


        scoreLabel = new JLabel("Score: ");
        scoreLabel.setBounds(860, 550, 200, 50);
        scoreLabel.setForeground(Color.YELLOW);
        scoreLabel.setFont(new Font("Tahoma", Font.BOLD, 30));

        timeLabel = new JLabel("Time: ");
        timeLabel.setBounds(860, 500, 200, 50);
        timeLabel.setForeground(Color.WHITE);
        timeLabel.setFont(new Font("Tahoma", Font.BOLD, 30));

        Image scaledwasdImage = new ImageIcon(getClass().getResource("wasd.png")).getImage().
getScaledInstance(200, 200, Image.SCALE_SMOOTH);
        JLabel wasdControls = new JLabel(new ImageIcon(scaledwasdImage));
        wasdControls.setBounds(50, 300, 200, 200);

        Image scaledarrowsImage = new ImageIcon(getClass().getResource("arrows.png")).getImage().
getScaledInstance(200, 150, Image.SCALE_SMOOTH);
        JLabel arrowControls = new JLabel(new ImageIcon(scaledarrowsImage));
        arrowControls.setBounds(1000, 280, 200, 200);


        setLayout(null);
        add(gameArea);
        add(nextBlockLabel1);
        add(nextBlockPanel1);

        add(nextBlockLabel2);
        add(nextBlockPanel2);

        add(scoreLabel);
        add(timeLabel);
        add(wasdControls);
        add(arrowControls);
        add(background);

        initControls();
        startGame();
    }
```

Construtor ของ Class GameFrame จะเป็นการจัดการ Layout ของหน้าเกมจริง พร้อมกับเริ่มเกมเลย

```
GameArea(int NumberOfLevel) {
        this.NumberOfLevel = NumberOfLevel;
        background = new Color[BOARD_HEIGHT][BOARD_WIDTH];
        grid = new Object[BOARD_HEIGHT][BOARD_WIDTH];
        blocks = new TetrisBlock[]{new ShapeI(),
                                   new ShapeJ(),
                                   new ShapeL(),
                                   new ShapeO(),
                                   new ShapeS(),
                                   new ShapeZ(),
                                   new ShapeT()};
        StarterBlock(NumberOfLevel);
    }
```

```
GameThread(GameArea gameArea, GameFrame gf ,int dalaytime)
{       this.dalaytime = dalaytime;
        this.gameArea = gameArea;
        this.gf = gf;
        music.setVolume(0.77f);
        music.loop();
    }
```

Constructor ของ Class GameArea จะเป็นการกำหนดค่าเริ่มต้นที่จำเป็นต่างๆในการเล่นเกม

Constructor ของ Class GameThread จะเป็นการกำหนดค่าเริ่มต้นที่จำเป็นต่างๆเพื่อใช้ใน Method run

```java
public HowToPlayDialog(JFrame parent) {
        super(parent, "How to Play", true);

        configureDialog(parent);

        JTable table = createTable();
        JScrollPane tableScrollPane = new JScrollPane(table);
        this.add(tableScrollPane, BorderLayout.CENTER);

        JLabel additionalText = createAdditionalTextLabel();
        JPanel textPanel = createTextPanel(additionalText);

        JButton closeButton = createCloseButton();
        JPanel buttonPanel = createButtonPanel(closeButton);
        JPanel bottomPanel = createBottomPanel(textPanel, buttonPanel);
        this.add(bottomPanel, BorderLayout.SOUTH);

        this.setVisible(true);
    }
```

```java
public ItemsGuideDialog(JFrame parent) {
        super(parent, "Items Guide", true);

        setDialogProperties(parent);

        JPanel itemsPanel = createItemsPanel();

        JScrollPane scrollPane = new JScrollPane(itemsPanel);
        this.add(scrollPane, BorderLayout.CENTER);

        JButton closeButton = new JButton("Close");
        closeButton.setFocusable(false);
        closeButton.addActionListener(e -> this.dispose());
        JPanel buttonPanel = new JPanel();
        buttonPanel.add(closeButton);
        this.add(buttonPanel, BorderLayout.SOUTH);

        this.setVisible(true);
    }
```

Constructor ของ Class HowToPlayDialog จะจัดการ Layout หน้าต่างโชว์วิธีการเล่น

Constructor ของ Class ItemGuideDialog จะจัดการ Layout หน้าต่างโชว์ข้อมูล Item

โดยทั้งสองจะอ่างอิงตำแหน่งจาก Class MyFrame

```java
public Item(int x, int y, String type) {
        if (x >= 0 && x < GameArea.BOARD_WIDTH) { this.x = x;}
        else { this.x = 0; }

        if (y >= 0 && y < GameArea.BOARD_HEIGHT) { this.y = y;
        } else { this.y = 0;}

        this.type = type;
        loadImage();
    }
```

```java
TetrisBlock(int [][] shape ){
        this.shape = shape;
        initShapes();
    }
```

Constructor ของ Class Item จะจัดการตำแหน่งการเกิดของ Item แต่ละชิ้นและแสดงผล

Constructor ของ Class TetrisBlock จะเก็บค่ารูปแบบของ Block เพื่อนำไปควบคุมต่อ

```java
public SoundEffect(String filePath) {
        try {
            URL audioSrc = getClass().getResource(filePath);
            if (audioSrc == null) {throw new IllegalArgumentException("Sound file not found: " + filePath
);}
            AudioInputStream audioStream = AudioSystem.getAudioInputStream(audioSrc);

            clip = AudioSystem.getClip();
            clip.open(audioStream);

            if (clip.isControlSupported(FloatControl.Type.MASTER_GAIN)) {
                volumeControl = (FloatControl) clip.getControl(FloatControl.Type.MASTER_GAIN);
            }
        } catch (Exception e) {}
    }
```

Constructor ของ Class SoundEffect จะรับข้อมูลที่อยู่ของไฟล์เสียงและจัดการการเล่นเสียง

- Encapsulation & Composition

```
public static final int BOARD_WIDTH = 11;
public static final int BOARD_HEIGHT = 16;
private static final int CELL_SIZE = 36;
private final Color[][] background;
private final Color[] RColor = {Color.BLACK,Color.GRAY,
Color.GREEN,Color.MAGENTA,Color.RED,Color.CYAN,Color.
PINK,Color.YELLOW};
private final Object[][] grid;

private TetrisBlock player1Block;
private TetrisBlock player2Block;
private TetrisBlock nextPlayer1Block;
private TetrisBlock nextPlayer2Block;

private final TetrisBlock[] blocks;
private final int NumberOfLevel;
```

โดยส่วนใหญ่ใน Class ต่างๆจะใช้การกำหนดตัวแปรเป็นแบบ Private แต่ก็มักจะมี Method เพื่อมาจัดการข้อมูลเหล่านี้แทน นอกจากนั้นแล้วในหลายๆ Class เองก็จะมีการ Composite Object จาก Class อื่นมาแล้วก็เก็บเป็นข้อมูล เพื่อที่จะสามารเช็ค, แก้ไข, ส่งออกข้อมูลไปยังส่วนอื่นได้อีก

- Polymorphism

```
public static void main(String[] args) {
        JFrame frame = new MyFrame();
        frame.setTitle("Tetris 2.0");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setResizable(false);
        frame.setSize(1280, 720);
        frame.setLocationRelativeTo(null);
        frame.setLayout(null);
        frame.setVisible(true);
    }
```

โดยใน Project นี้จะไม่ได้ใช้หลักการการพร้องรูปมากนักแต่ก็พอมีอยู่ตอนเริ่มต้นโปรแกรมโดยจะใช้ตัวแปร frame ชนิด JFrame ชี้ไปยังวัตถุที่สร้างจากสับคลาสชนิด MyFrame

- Inheritance

```
public class MyFrame extends JFrame
```

```
public class HowToPlayDialog extends JDialog
```

```
public class GameThread extends Thread
```

```
public class GameArea extends JPanel
```

```
public class ShapeI extends TetrisBlock
```

โดยในตัวอย่างนี้คือ หลายๆ Class จะได้มีการ extend จาก Class ที่มีอยู่แล้วใน Java แต่บาง Class จะเป็นการ Inheritance ถึง Class ที่สร้างเองด้วยเช่นกัน

- Abstract

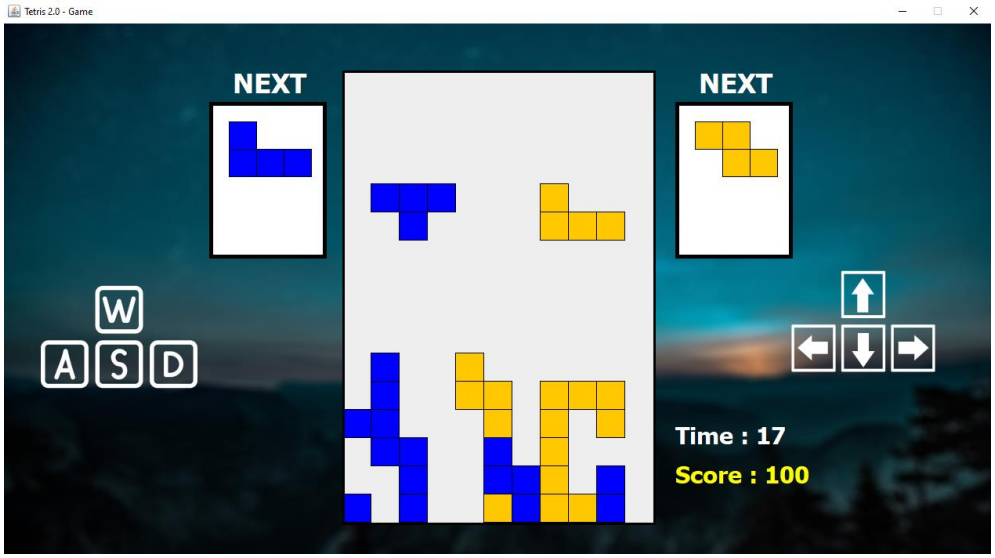
```java
@Override
    public void run() {
        gf.updateScore(score);
        startTimer();


        int blockCount = 0;
        while (gameRunning) {
            gameArea.spawnBlocks();
            gf.repaintNextBlockPanel();

            boolean canMove1 = true;
            boolean canMove2 = true;

            blockCount++;
            if (blockCount >= 3) {
                spawnItem();
                blockCount = 0;
            }

            while (canMove1 || canMove2) {
                gameArea.moveItemsDown();
                checkItemCollision(gameArea.getCurrentBlockPosition1());
                checkItemCollision(gameArea.getCurrentBlockPosition2());

                if (canMove1) { canMove1 = gameArea.moveBlockDown1();}
                if (canMove2) { canMove2 = gameArea.moveBlockDown2();}

                sleep(dalaytime);

                if (!canMove1) {gameArea.moveBlockToBackground1();}
                if (!canMove2) {gameArea.moveBlockToBackground2();}
            }

            score += gameArea.clearLines() * 100;
            gf.updateScore(score);

            if (gameArea.isBlockOutOfBounds1() || gameArea.isBlockOutOfBounds2()) {
                gameRunning = false;
                music.stop();
                gf.showGameOverPopup(score, time);
                break;
            }
        }
    }
```

```java
@Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        drawBackground(g);
        drawBlock1(g);
        drawBlock2(g);
        drawItems(g);
    }
```

โดยในการใช้งานเรื่อง Abstract ใน Project นี้นั้นไม่ได้มีการสร้าง Abstract Class / Method ขึ้นมาเอง แต่จะใช้งาน Abstract Method ที่มีอยู่แล้วในภาษา Java

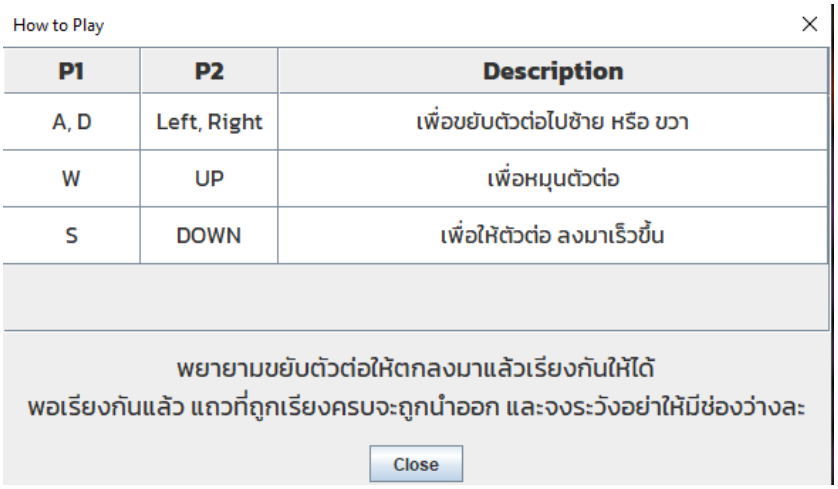
2.5 ส่วนประกอบ GUI

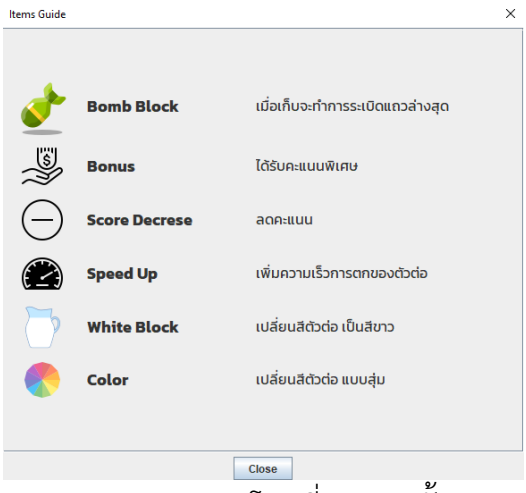
โดยหลักๆแล้ว Component ต่างๆในหน้าเมนูเกมนั้นจะถูกจัดการด้วย Class MyFrame ดั่งที่ได้เห็นแล้วใน Constructor โดยก็จะมีการส่วนประกอบดังนี้คือ Background / Logo ที่เป็น ImageIcon แล้วใส่ใน JLabel background แล้วก็มี JButton 4 ตัวคือ startButton, howToPlayButton, itemsGuideButton, exitButton JComboBox<String> 2 ตัวคือ difficultyComboBox, levelComboBox;



ในหน้าเกมหลักนั้นจะถูกจัดการโดย Class GameFrame และ Class GameArea โดย GameFrame จะจัดการส่วนของพื่นหลังโดยก็จะมี JLabel 2 ตัวคือ scoreLabel,timeLabel และ JPanel 2 ตัวคือ nextBlockPanel1,nextBlockPanel2 และก็ตัวจะการรูปภาพพื่นหลัง และ logo wasd , arrow โดย ImageIcon

และส่วนของ GameArea ก็จะจัดการในพื่นที่เกมหลังตรงกลางหน้าจอที่จะก็มาใช้งานโดย JPanel จะทำงานด้วยการวาดบน Graphics ทั้งหมดจากการ Override paintComponent





ในส่วนของ How to Play และ Items Guide จะใช้เป็นตัวของการ extends JDialog โดยที่ Class ทั้งสองจะอ่างอิงตำแหน่งจาก Class MyFrame เพื่อที่จะแสดงผลตรงกลางโปรแกรม

Class HowToPlayDialog นั้นก็จะใช้เป็น JTable เพื่อความง่ายต่อการอ่าน และ JLabel สำหรับคำแนะนำการเล่น และ JButton ปิดหน้าต่าง

Class ItemsGuideDialog นั้นก็จะใช้เป็น JPanel โดยจะดึงค่าเป็น Array Object และก็จะมีการกำหลังเป็น Grid และ JButton ปิดหน้าต่าง

2.6 Event handling

```
private static class Listener implements ActionListener {
        private final MyFrame frame;
        Listener(MyFrame frame) { this.frame = frame; }

        @Override
        public void actionPerformed(ActionEvent e) {
            if (e.getSource() == frame.startButton) {
                String selectedDifficulty = (String) frame.difficultyComboBox.getSelectedItem();
                switch (selectedDifficulty) {
                    case "Easy": frame.dalaytime = 1000;  break;
                    case "Normal": frame.dalaytime = 500; break;
                    case "Hard":frame.dalaytime = 250;    break;
                }

                String selectedLevel = (String) frame.levelComboBox.getSelectedItem();
                switch (selectedLevel) {
                    case "Level 1": frame.NumberOfLevel = 0; break;
                    case "Level 2": frame.NumberOfLevel = 1; break;
                    case "Level 3": frame.NumberOfLevel = 2; break;
                    case "Level 4": frame.NumberOfLevel = 3; break;
                    case "Level 5": frame.NumberOfLevel = 4; break;
                    case "Level 6": frame.NumberOfLevel = 5; break;
                }

                frame.dispose();
                SoundEffect buttonClickSound = new SoundEffect("/item/Click.wav");
                buttonClickSound.play();

                GameFrame gameFrame = new GameFrame(frame.dalaytime,frame.NumberOfLevel);
                gameFrame.setTitle("Tetris 2.0 - Game");
                gameFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                gameFrame.setResizable(false);
                gameFrame.setSize(1280, 720);
                gameFrame.setLocationRelativeTo(null);
                gameFrame.setLayout(null);
                gameFrame.setVisible(true);
            } else if (e.getSource() == frame.howToPlayButton) {
                SoundEffect buttonClickSound = new SoundEffect("/item/Click.wav");
                buttonClickSound.play();
                new HowToPlayDialog(frame);
            } else if (e.getSource() == frame.itemsGuideButton) {
                SoundEffect buttonClickSound = new SoundEffect("/item/Click.wav");
                buttonClickSound.play();
                new ItemsGuideDialog(frame);
            } else if (e.getSource() == frame.exitButton) {
                SoundEffect buttonClickSound = new SoundEffect("/item/Click.wav");
                buttonClickSound.play();
                System.exit(0);
            }
        }
    }
```

```
@Override
    public void run() {
        gf.updateScore(score);
        startTimer();


        int blockCount = 0;
        while (gameRunning) {
            gameArea.spawnBlocks();
            gf.repaintNextBlockPanel();

            boolean canMove1 = true;
            boolean canMove2 = true;

            blockCount++;
            if (blockCount >= 3) {
                spawnItem();
                blockCount = 0;
            }

            while (canMove1 || canMove2) {
                gameArea.moveItemsDown();
                checkItemCollision(gameArea.getCurrentBlockPosition1());
                checkItemCollision(gameArea.getCurrentBlockPosition2());

                if (canMove1) { canMove1 = gameArea.moveBlockDown1();}
                if (canMove2) { canMove2 = gameArea.moveBlockDown2();}

                sleep(dalaytime);

                if (!canMove1) {gameArea.moveBlockToBackground1();}
                if (!canMove2) {gameArea.moveBlockToBackground2();}
            }

            score += gameArea.clearLines() * 100;
            gf.updateScore(score);

            if (gameArea.isBlockOutOfBounds1() || gameArea.isBlockOutOfBounds2()) {
                gameRunning = false;
                music.stop();
                gf.showGameOverPopup(score, time);
                break;
            }
        }
    }
```

```
private void initControls() {
        InputMap im = this.getRootPane().getInputMap();
        ActionMap am = this.getRootPane().getActionMap();

        im.put(KeyStroke.getKeyStroke('d'), "right2");
        im.put(KeyStroke.getKeyStroke('a'), "left2");
        im.put(KeyStroke.getKeyStroke('w'), "rotate2");
        im.put(KeyStroke.getKeyStroke('s'), "drop2");

        am.put("right2", new AbstractAction() {
            @Override
            public void actionPerformed(ActionEvent e) {
                gameArea.moveBlockRight1();
            }
        });
        am.put("left2", new AbstractAction() {
            @Override
            public void actionPerformed(ActionEvent e) {
                gameArea.moveBlockLeft1();
            }
        });
        am.put("rotate2", new AbstractAction() {
            @Override
            public void actionPerformed(ActionEvent e) {
                gameArea.rotateBlock1();
            }
        });
        am.put("drop2", new AbstractAction() {
            @Override
            public void actionPerformed(ActionEvent e) {
                gameArea.dropBlock1();
                SoundEffect drop = new SoundEffect("/item/drop.wav");
                drop.setVolume(0.7f);
                drop.play();
            }
        });

        im.put(KeyStroke.getKeyStroke("RIGHT"), "right1");
        im.put(KeyStroke.getKeyStroke("LEFT"), "left1");
        im.put(KeyStroke.getKeyStroke("UP"), "rotate1");
        im.put(KeyStroke.getKeyStroke("DOWN"), "drop1");

        am.put("right1", new AbstractAction() {
            @Override
            public void actionPerformed(ActionEvent e) {
                gameArea.moveBlockRight2();
            }
        });
        am.put("left1", new AbstractAction() {
            @Override
            public void actionPerformed(ActionEvent e) {
                gameArea.moveBlockLeft2();
            }
        });
        am.put("rotate1", new AbstractAction() {
            @Override
            public void actionPerformed(ActionEvent e) {
                gameArea.rotateBlock2();
            }
        });
        am.put("drop1", new AbstractAction() {
            @Override
            public void actionPerformed(ActionEvent e) {
                gameArea.dropBlock2();
                SoundEffect drop = new SoundEffect("/item/drop.wav");
                drop.setVolume(0.7f);
                drop.play();
            }
        });
    }
```

เกมนี้จะได้มีการใช้งาน ActionListener กับตัวของ JButton ทั้ง 4 ตัวใน Class MyFrame และก็ยังจะมีการใช้งานตัวของ AbstractAction ที่เป็นแบบ Anonymous ด้วยโดยจะเป็นการกำหนดการควบคุมให้กับ WASD , ARROW KEY เพื่อควบคุมตัวต่อ และในส่วนของการควบคุมเวลาจะใช้ Thread ในการควบคุมโดยความการ Delay จะขึ้นอยู่กับความยากที่ผู้เล่นเลือกเอง

2.7 อัลกอริทึมที่สำคัญ

โดยหลักๆแล้วอัลกอริทึมตัวสำคัญจะเป็นการจัดการตัวของ Block ของทั้งสองผู้เล่น โดยก็จะมีการขยับตัวต่อลงมา การหมุนตัวต่อ การเช็คว่าได้ไปชนอะไรหรือป่าวเป็นต้น โดยรูปแบบการเขียนอัลกอริทึมในเกมนี้จะเป็นแบบ Method ทำอย่างทีละอย่าง โดยจะมีตัวอย่างคร่าวๆดังต่อไปนี้

`public boolean isBlockOutOfBounds1(){`
`  return player1Block != null && player1Block.getY() < 0;`
`  }`
	เป็นการเช็คการเกินขอบเขตของเกมเพื่อจบเกม

`public boolean moveBlockDown1(){`
`  if(checkBottom1() == false){return false;}`
 	
`  player1Block.moveDown();`
`  repaint();`
`  return true;`
`  }`
	เป็นการขยับตัวต่อลง โดยจะใช้การ repaint

`public void moveBlockRight1(){`
`  if(player1Block == null) return;`
`  if(!checkRight1()) return;`
`  if(!checkBottom1()) return;`
`  player1Block.moveRight();`
`  if(checkBlockCollision(player1Block, player2Block)) {`
`  player1Block.moveLeft();`
`  return;`
`  }`
`  repaint();`
`  }`
	เป็นการขยับตัวต่อไปขวา โดยจะใช้การ repaint

`public void moveBlockLeft1(){`
`  if(player1Block == null) return;`
`  if(!checkLeft1()) return;`
`  if(!checkBottom1()) return;`
`  player1Block.moveLeft();`
`  if(checkBlockCollision(player1Block, player2Block)) {`
`  player1Block.moveRight();`
`  return;`
`  }`
`  repaint();`
`  }`
	เป็นการขยับตัวต่อไปซ้าย โดยจะใช้การ repaint

`public void dropBlock1(){`
`  if(player1Block == null) return;`
`  while (checkBottom1()) {`
`  player1Block.moveDown();`
`  }`
`  moveBlockToBackground1();`
`  repaint();`
`  }`
	เป็นการขยับตัวต่อลงจนสุดก่อนแล้ว repaint

`public void rotateBlock1(){`
`  if(player1Block == null) return;`
`  if(player1Block.getLeftEdge() < 0) player1Block.setX(0);`
`  if(player1Block.getRightEdge() >= BOARD_WIDTH) player1Block.setX(BOARD_WIDTH - player1Block.getWidth());`
`  if(player1Block.getBottomEdge() >= BOARD_HEIGHT) player1Block.setY(BOARD_HEIGHT - player1Block.getHeight());`
`  if(!checkLeft1()) return;`
`  if(!checkRight1()) return;`
`  if(!checkBottom1()) return;`
`  player1Block.rotate();`
`  repaint();`
`  }`
	เป็นการหมุนตัวต่อ โดยจะใช้การ repaint

	เป็นเช็คเพื่อขยับตัวต่อลงมาว่าได้หรือไม่ โดยหลักๆคือจะวน loop เช็คกับ background ว่าว่างหรือไม่
	เป็นเช็คเพื่อขยับตัวต่อไปซ้ายว่าได้หรือไม่ โดยหลักๆคือจะวน loop เช็คกับ background ว่าว่างหรือไม่
	เป็นเช็คเพื่อขยับตัวต่อไปขวาว่าได้หรือไม่ โดยหลักๆคือจะวน loop เช็คกับ background ว่าว่างหรือไม่
	เป็นการเช็คการชนกันของ Block ทั้งสองของผู้เล่น โดยจะอิงจากตำแหน่ง ว่าถ้าตำแหน่งไม่ได้จะชนกันก็สามารถขยับได้ปกติ แต่ถ้ามันจะชนก็จะไม่เกิดการขยับ

	เมื่อ Block มาจนถึงด้านล่างแล้วเราจะเปลี่ยนBlock นั้นให้กลายเป็น Background เพื่อไปใช้ในการเช็คการขยับของ Method อื่นตามเดิม
	เป็นการวาดตัวของ Block ผู้เล่นตลอดโดยคำสั่งนี้จะถูกใช้งานใน protected void paintComponent(Graphics g) หรือเมื่อ Method อื่นๆ มีการ repaint นั้นเอง

หัวข้อที่ 3

สรุป

3.1 ปัญหาที่พบระหว่างการพัฒนา

- Block ซ้อนกัน , หมุน Block ทะลุขอบ , เวลาไม่หยุด , การจัดการเกมการเล่นสองคนที่ยุ่งยาก เพราะถึงแม้จะพอมีเนื้อหาเกี่ยวกับเกม Teris แต่ก็เป็นแค่แบบ Basic จึงทำให้เสียเวลานั้งคิดวิธีการเอง

3.2 จุดเด่นของโปรแกรม

- เป็นการเปลี่ยนแนวคิดเกม Teris ที่ส่วนมากจะเป็นการเล่นคนเดียว หรือ แข่งขันกัน โดยเปลี่ยนให้เป็นการร่วมมือกันแทน , มีการเลือกระดับความยาก , มีเสียงประกอบที่ชวนคิดถึง แต่ก็ไม่เหมือนเดิม

3.3 คำแนะนำสำหรับผู้สอน

- อยากจะให้ส่ง code เฉลยที่เป็นแบบฝึกหัดมาให้ศึกษาหน่อยครับ เพื่อที่จะได้รู้ข้อผิดพลากของตนเองเลย