รายงานโครงงาน

วิชา Object-Oriented Programming

เกม Sudoku

จัดทำโดย

1. นายธัญธร จันทร์เรียง 58070058

2. นายธนชาติ อ่ำไร่ขิง 59070064

3. นายธชย อินวะษา 59070185

เสนอ

ผศ.ดร. ธนิศา นุ่นนนท์

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา

06016211 Object-Oriented Programming

สาขาเทคโนโลยีสารสนเทศ คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ภาคเรียนที่ 1 ปีการศึกษา 2561

บทคัดย่อ

ปัจจุบันเทคโนโลยีได้เข้ามามีบทบาทในชีวิตประจำวันมากขึ้น ไม่ว่าจะเป็นทั้งด้านการทำงาน ด้านการทำธุรกรรมต่างๆ ด้านการเรียน รวมไปถึงด้านความบันเทิงเพื่อผ่อนคลายความเครียดและความเหนื่อยล้า ทำให้มีการการพัฒนาแอปพลิเคชั่น และเกมต่างๆบนคอมพิวเตอร์ขึ้นมาอย่างมากมาย ผู้จัดทำจึงมีความคิดที่อยากจะพัฒนาเกมที่ให้ทั้งความสนุกเพลินเพลิน ช่วยผ่อนคลายความเครียด รวมไปถึงช่วยในการพัฒนากระบวนการทางด้านความคิด สมาธิ และสติปัญญา จึงเกิดเป็นโครงงานชิ้นนี้ขึ้นมา

ผู้จัดทำ

สารบัญ

บทที่ 1

บทนำ

ที่มาและความสำคัญของโครงงาน

เมื่อคอมพิวเตอร์เข้ามาเป็นส่วนหนึ่งในชีวิตประจำวันมากขึ้น ทำให้มีการพัฒนาเกม และแอพพลิเคชั่นต่างๆ เพิ่มมากขึ้นเช่นกัน คณะผู้จัดทำจึงเล็งเห็นว่าสามารถนำความรู้ที่ได้จากการเรียนในวิชา Object-Oriented Programming มาพัฒนาเกมที่เล่นง่าย และมีความสนุกสนาน เพื่อเป็นทางเลือกให้แก่ผู้ใช้งาน และผู้ที่สนใจต่อไป

วัตถุประสงค์ของโครงงาน

1. เพื่อนำความรู้เบื้องต้นจากภาษาจาวา และหลักการของ Object-Oriented Programming มาประยุกต์ใช้ และต่อยอดในการพัฒนาเกม
2. เพื่อพัฒนาเกมที่มีความสนุกสนานเพลิดเพลิน แถมยังช่วยในการพัฒนากระบวนการคิด
3. ช่วยให้รู้จักการแก้ปัญหาอย่างเป็นระบบ และเป็นขั้นเป็นตอน

ขอบเขตการศึกษาค้นคว้า

- ภาษาที่ใช้ในการพัฒนาคือ ภาษาจาวา
- ศึกษาเพิ่มเติมในส่วนของโปรแกรม Eclipse และ Adobe Photoshop

ประโยชน์ที่คาดว่าจะได้รับ

- มีทักษะในการคิด เขียน วิเคราะห์และจัดการกับปัญหา ในการเขียนอัลกอริทึมของภาษาจาวาเพิ่มมากขึ้น รวมทั้งเข้าใจหลักการของ Object-Oriented Programming มากขึ้น
- เกมที่สร้างขึ้นมามีความสนุกสนานเพลิดเพลิน ช่วยผ่อนคลายความเครียดได้
- มีการซึมซับ และรู้จักการแก้ปัญหาอย่างเป็นระบบ และเป็นขั้นเป็นตอน

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

ในการจัดทำโครงงานเรื่อง เกม Sudoku ในครั้งนี้ ผู้จัดทำได้มีการศึกษาข้อมูลสำหรับนำมาจัดทำโครงงานดังนี้

1. เกม Sudoku (ลักษณะของเกม , กติกาและวิธีการเล่น)
2. ภาษา Java

เกม Sudoku

Sudoku อ่านออกเสียง “ซู – โด – กุ” คือเกมปริศนาตารางตัวเลข ที่ผู้เล่นจะต้องเติมตัวเลขลงในช่องว่างของตาราง โดยจะต้องใช้ตัวเลข ไม่ให้ซ้ำกัน และใช้ตัวเลขแต่ละตัวได้เพียงครั้งเดียวเท่านั้น ทั้งในทุกแถวของแนวตั้ง แนวนอน ผู้เล่นจะต้องใช้หลักเหตุและผลหรือตรรกะ (logic) ในการไขปริศนานี้โดยดูจากข้อมูลตัวเลขที่ให้มา

วิธีการเล่น ผู้เล่นต้องเลือกใส่ หมายเลขตั้งแต่ เลข 1 ถึงเลข 9 สำหรับตาราง 9 × 9 ช่อง และเลข 1 ถึงเลข 16 สำหรับตาราง 16 × 16 ช่อง โดยมีเงื่อนไขว่าในแต่ละแถวและแต่ละหลักตัวเลขต้องไม่ซ้ำกัน ตารางซูโดะกุที่มี 9 × 9 ช่องจะประกอบจากตารางย่อย 9 ตาราง ในลักษณะ 3 × 3 แบ่งแยกกันโดยเส้นหนา และในแต่ละตารางย่อยจะต้องมีตัวเลข 1 ถึง 9 เช่นเดียวกัน ส่วนตาราง 16 × 16 ก็จะประกอบด้วยตาราง 9 ตารางเช่นกันแต่ภายในจะมีตารางย่อยเป็น 4 × 4 ช่องและในตารางต้องมีเลข 1 ถึงเลข 16 เช่นเดียวกัน

ภาษาจาวา

Java เป็นภาษาที่ใช้ในการเขียนคำสั่งสั่งงานคอมพิวเตอร์ ซึ่งพัฒนาขึ้นโดยบริษัท ซันไมโครซิสเต็มส์ จำกัด (Sun Microsystems Inc.) ในปี ค.ศ. 1991 เป็นส่วนหนึ่งของโครงการวิจัยเพื่อพัฒนาซอฟต์แวร์ สำหรับอุปกรณ์อิเล็กทรอนิคส์ต่างๆ เช่น โทรทัศน์ โทรศัพท์มือถือ โดยมีเป้าหมายการทำงานเชื่อมต่อกับอุปกรณ์ฮาร์ดแวร์ต่างๆ ได้อย่างกว้างขวาง และมีประสิทธิภาพ ใช้เวลาน้อย รวดเร็วในการพัฒนาโปรแกรม และสามารถเชื่อมต่อไปยังแพล็ตฟอร์ม (Platform) อื่นๆ ได้ง่าย Java เป็นภาษาสำหรับเขียนโปรแกรมภาษาหนึ่งที่มีลักษณะสนับสนุนการเขียนโปรแกรมเชิงวัตถุ (OOP : Object-Oriented Programming) ที่ชัดเจน โปรแกรมต่างๆ ถูกสร้างภายในคลาส (Class) โปรแกรมเหล่านั้นเรียกว่า Method หรือ Behavior โดยปกติจะเรียกแต่ละ Class ว่า Object โดยแต่ละ Object มีพฤติกรรมมากมาย โปรแกรมที่สมบูรณ์จะเกิดจากหลาย Object หรือหลาย Class มารวมกัน โดยแต่ละ Class จะมี Method หรือ Behavior แตกต่างกันไป

บทที่ 3
วิธีการดำเนินการ

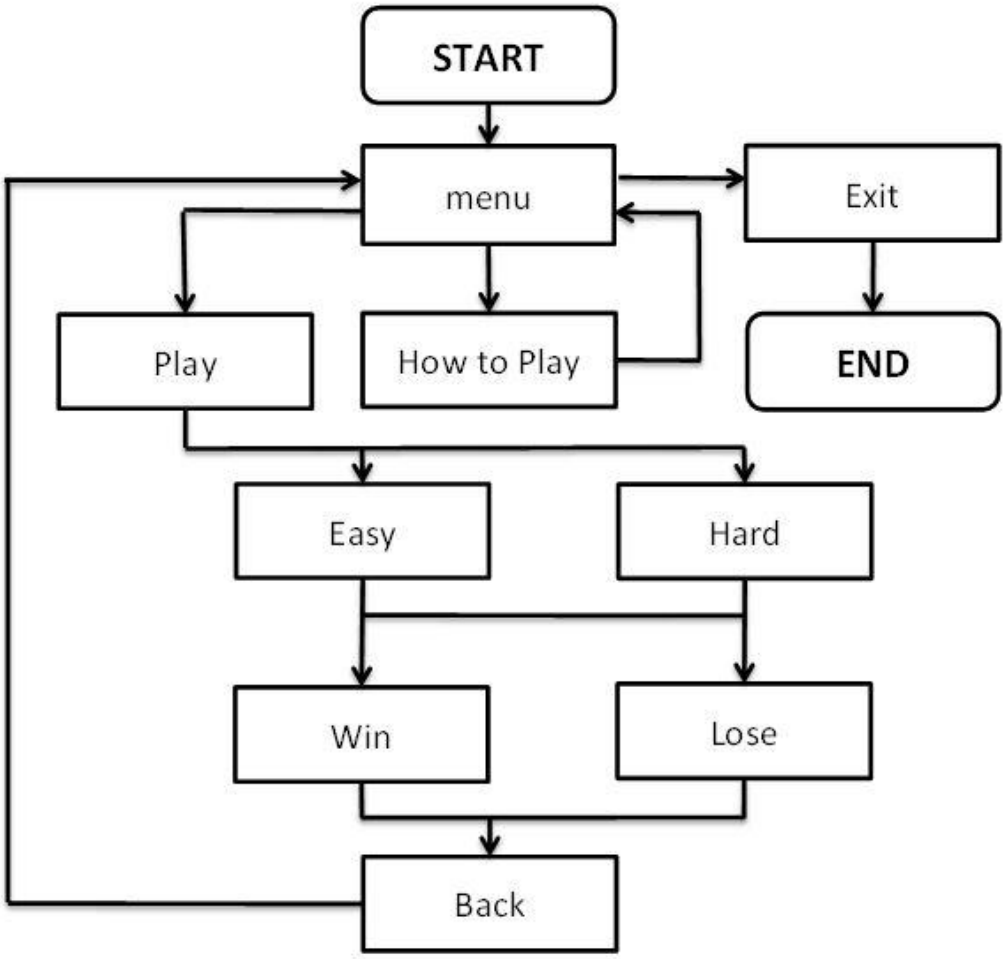
เครื่องมือที่ใช้ในการจัดทำ

- Eclipse
- Adobe Photoshop

ขั้นตอนการดำเนินงาน

1. วางแผน ระดมความคิด สรุป โครงงานที่จัดทำ
2. ศึกษาเครื่องมือในการจัดทำโครงงาน อาทิ เช่น Lib GDX
3. เขียนผังงาน
4. สรุปขอบเขตและภาพรวม
5. ลงมือดำเนินงานที่ได้รับมอบหมาย

หลักการทำงานของเกม



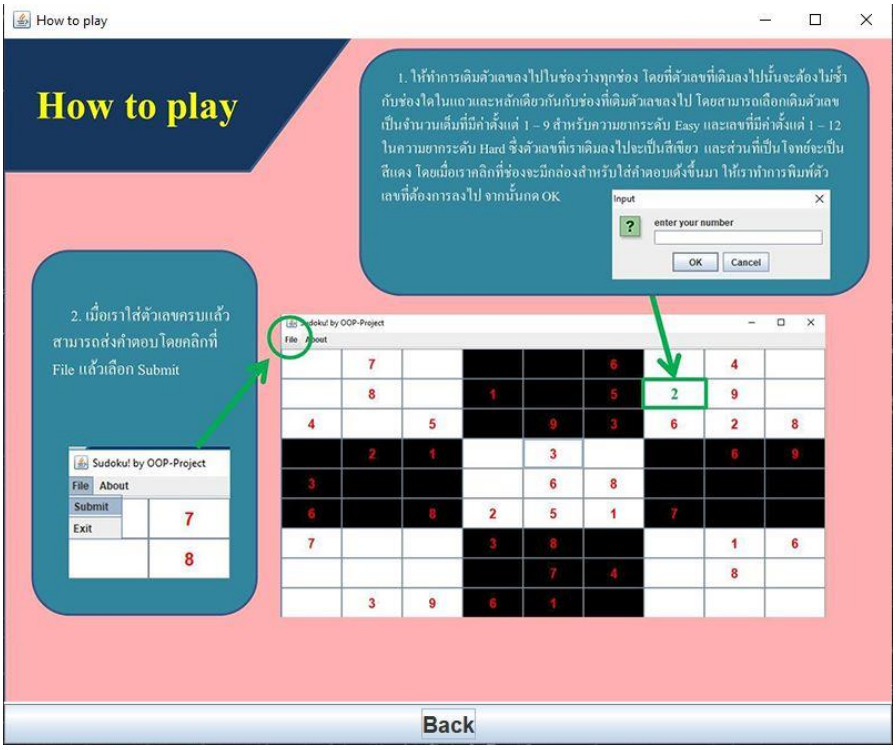
วิธีการเล่นเกม

1.กดเปิดเกม

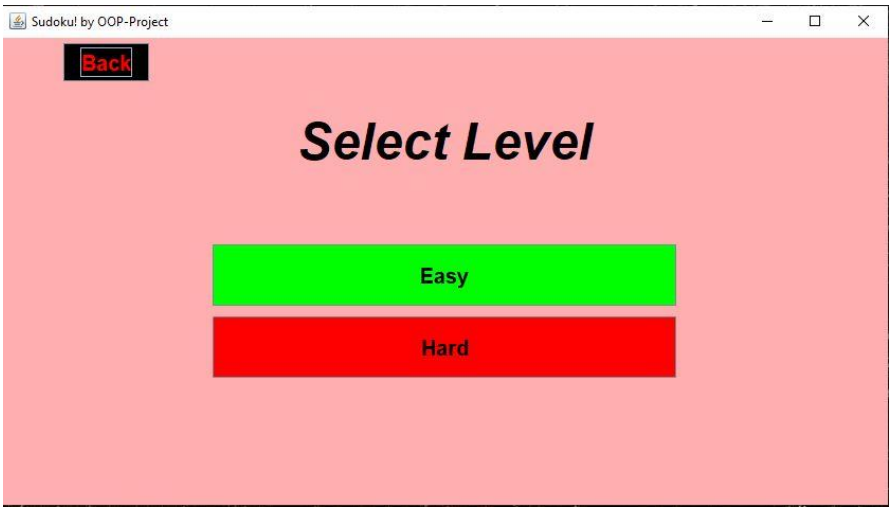
2.เมื่อเปิดตัวเกมขึ้นมา หน้าแรกจะให้ใส่ชื่อผู้เล่น

3.หลังจากใส่ชื่อผู้เล่นเสร็จ สามารถเลือกกด Play เพื่อเริ่มเล่น หรือHow to Play เพื่อดูวิธีการเล่น หรือ Exit เพื่อออกจากเกม

4.หากเลือกกด How to Play จะแสดงหน้าต่างวิธีการเล่นเกม บอกวิธีการเล่นเกม และหากกด Back จะย้อนกลับไปหน้าแรก

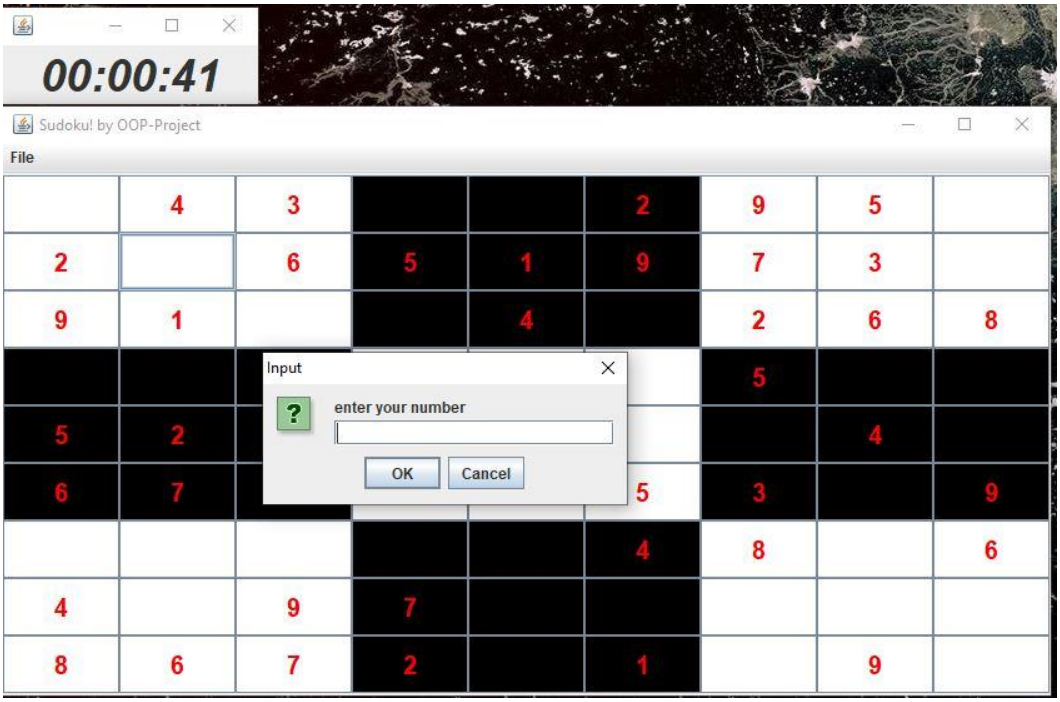


5.หากเลือกกด Play จะแสดงหน้าต่าง Select Level ซึ่งภายในเกมจะมีให้เลือกระดับความ ยากง่าย 2 ระดับ คือ ง่าย (Easy) และยาก (Hard)

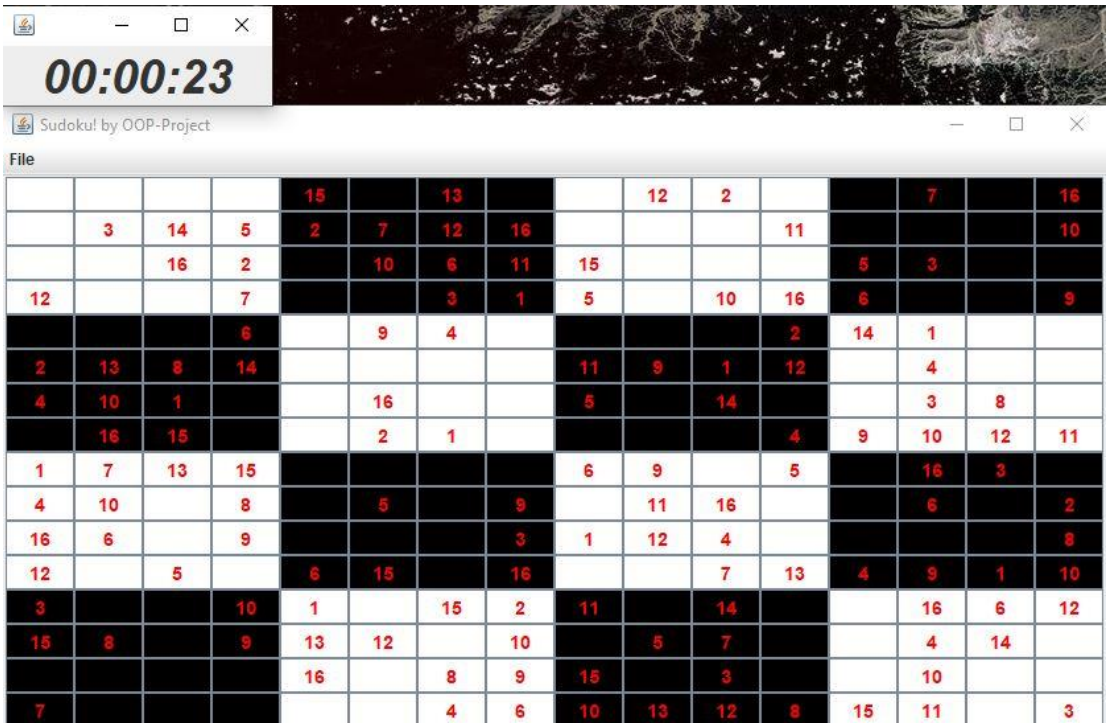


6. เมื่อเลือกระดับแล้วก็จะเข้าสู่หน้าเล่นเกมโดยเกมจะสุ่มตัวเลขมาให้ ผู้เล่นจะต้องเติมตัวเลขลงในช่องว่างของตาราง โดยจะต้องใช้ตัวเลข ไม่ให้ซ้ำกัน และใช้ตัวเลขแต่ละตัวได้เพียงครั้งเดียวเท่านั้น ทั้งในทุกแถวของแนวตั้ง แนวนอน ผู้เล่นจะต้องใช้หลักเหตุและผลหรือตรรกะ (logic) ในการไขปริศนานี้โดยดูจากข้อมูลตัวเลขที่ให้มา และจะมีเวลาที่ใช้ไปทั้งหมดจนกว่าจะจบเกม

7. หากผู้เล่นเลือกระดับง่าย (Easy) ผู้เล่นต้องเลือกใส่ หมายเลขตั้งแต่ เลข 1 ถึงเลข 9 โดยมีเงื่อนไขว่าในแต่ละแถวและแต่ละหลักตัวเลขต้องไม่ซ้ำกัน ตารางซูโดะกุที่มี 9 × 9 ช่องจะประกอบจากตารางย่อย 9 ตาราง ในลักษณะ 3 × 3 แบ่งแยกกันโดยเส้นหนา และในแต่ละตารางย่อยจะต้องมีตัวเลข 1 ถึง 9 เมื่อผู้เล่นกดช่องที่จะใส่เลข จะแสดงปอปอัพขึ้นมาให้ใส่ตัวเลข



8.หากผู้เล่นเลือกระดับยาก (Hard) ผู้เล่นต้องเลือกใส่ หมายเลขตั้งแต่ เลข 1 ถึงเลข 16 โดยมีเงื่อนไขว่าในแต่ละแถวและแต่ละหลักตัวเลขต้องไม่ซ้ำกัน ตารางซูโดะกุที่มี 16× 16 ช่องจะประกอบจากตารางย่อย 9 ตาราง ในลักษณะ 4 × 4 แบ่งแยกกันโดยเส้นหนา และในแต่ละตารางย่อยจะต้องมีตัวเลข 1 ถึง 16 เมื่อผู้เล่นกดช่องที่จะใส่เลข จะแสดงปอปอัพขึ้นมาให้ใส่ตัวเลข



				15		13			12	2			7		16
	3	14	5	2	7	12	16				11				10
		16	2		10	6	11	15				5	3		
12			7			3	1	5		10	16	6			9
			6		9	4					2	14	1		
2	13	8	14					11	9	1	12		4		
4	10	1			16			5		14			3	8	
	16	15			2	1					4	9	10	12	11
1	7	13	15					6	9		5		16	3	
4	10		8		5		9		11	16			6		2
16	6		9				3	1	12	4					8
12		5		6	15		16			7	13	4	9	1	10
3			10	1		15	2	11		14			16	6	12
15	8		9	13	12		10		5	7			4	14	
				16		8	9	15		3			10		
7						4	6	10	13	12	8	15	11		3

9.เมื่อผู้เล่นใส่ตัวเลขจนครบแล้ว ให้กดเมนู File >> Submit

10.หากผู้เล่นชนะ จะแสดงดังภาพ

11.หาผู้เล่นแพ้ จะแสดงดังภาพ

บทที่ 4

ผลการดำเนินงาน

ผลการดำเนินงาน

จากการศึกษากระบวนการและวิธีการทั้งหมดที่กล่าวมาข้างต้น ทำให้ผู้จัดทำสามารถดำเนินงานได้ตามแผนงานที่วางไว้ และได้เกม Sudoku ที่มีความสนุกสนานน่าเล่นตามที่วางเป้าหมายไว้

การนำไปใช้

เกมที่ได้สามารถใช้งานได้จริง เล่นได้จริง และมีความสนุกสนามเพลิดเพลินตามที่หวัง

บทที่ 5

สรุปผลและข้อเสนอแนะ

ผลที่ได้รับ

1. ได้เรียนรู้วิธีการสร้างเกมโดยภาษาจาวา หลักการของ Object-Oriented Programming และไลบารีอื่นๆ จากการสืบค้นเพิ่มเติม
2. ผู้เล่นได้เล่นเกมที่สนุกสนานเพลิดเพลิน
3. เกมที่ได้ช่วยให้รู้จักการวางแผน และดำเนินการแก้ปัญหาอย่างเป็นขั้นเป็นตอน

ปัญหาและอุปสรรค

- มีการเปลี่ยนแปลงเนื้อหาโครงงาน ทำให้เวลาในการทำงานเหลือน้อยลง
- ความรู้และความเชี่ยวชาญในการเขียนโปรแกรมยังมีข้อบกพร่องอยู่บ้าง ทำให้ทำงานได้ช้ากว่าที่ควรจะเป็น
- ใช้เวลาในการศึกษาเพิ่มเติมเกี่ยวกับการเขียนโปรแกรมเยอะพอสมควร

ข้อเสนอแนะและแนวทางในการพัฒนา

- ยังไม่สามารถเล่นผ่าน Smart Phone ได้
- Level ที่มีให้เลือกเล่นน้อย
- อยากให้มีเกมในรูปแบบอื่นที่คล้ายกันรวมอยู่ด้วย เช่นเกมเติมคำ

อ้างอิง

blog.dechathon.com เกม Sudoku และวิธีการเล่น

stackoverflow.com กระบวนการคิด และหลักการเขียนโปรแกรมด้วยภาษา Java

ภาคผนวก

```
import java.awt.*;

import java.awt.event.*;

import javax.swing.*;


public class Main implements ActionListener {


                private JFrame frame;

                private JPanel main, panel, btPanel, left, right, bottom;

                private JButton play, exit, tutorial, back;

                private JLabel name;

                private static String player;


                public Main() {

                 frame = new JFrame("Sudoku! by OOP-Project");

                 main = new JPanel(new BorderLayout());

                 panel = new JPanel(new GridLayout(2, 1));

                 btPanel = new JPanel(new GridLayout(3, 1, 10, 10));

                 left = new JPanel();

                 right = new JPanel();
```

```
bottom = new JPanel();

play = new JButton("Play");

exit = new JButton("Exit");

tutorial = new JButton("How to Play");


name = new JLabel("Sudoku!");


play.addActionListener(this);

tutorial.addActionListener(this);

exit.addActionListener(this);


name.setFont(new Font("ARIALBD", Font.ITALIC | Font.BOLD, 56));

name.setForeground(Color.black);

name.setHorizontalAlignment(SwingConstants.CENTER);


panel.setBackground(Color.pink);

btPanel.setBackground(Color.pink);

main.setBackground(Color.pink);

left.setBackground(Color.pink);

right.setBackground(Color.pink);
```

```
bottom.setBackground(Color.pink);


play.setFont(new Font("ARIALBD", Font.BOLD, 20));

play.setForeground(Color.black);

play.setBackground(Color.green);

tutorial.setFont(new Font("ARIALBD", Font.BOLD, 20));

tutorial.setForeground(Color.black);

tutorial.setBackground(Color.yellow);

exit.setFont(new Font("ARIALBD", Font.BOLD, 20));

exit.setForeground(Color.black);

exit.setBackground(Color.red);


btPanel.add(play);

btPanel.add(tutorial);

btPanel.add(exit);


panel.add(name);

panel.add(btPanel);
```

```
 left.setPreferredSize(new Dimension(200, 480));

 right.setPreferredSize(new Dimension(200, 480));

 bottom.setPreferredSize(new Dimension(854, 50));


 main.add(panel, BorderLayout.CENTER);

 main.add(left, BorderLayout.WEST);

 main.add(right, BorderLayout.EAST);

 main.add(bottom, BorderLayout.SOUTH);


 frame.add(main);

 frame.setSize(854, 480);

 frame.setLocationRelativeTo(null);

 frame.setVisible(true);


}


public static String getName() {

 return player;

}
```

```
public void setName() {
 while(true) {
        player = "";
        if(player.equals("")) {
                String inp = JOptionPane.showInputDialog(null, "enter your
name");
                if(inp != null && inp.length() >= 1) {
                        player = inp;
                        break;
                }
        }
 }
}

public void actionPerformed(ActionEvent ev) {
 String cmd = ev.getActionCommand();
 if(cmd.equals("Play")) {
        new Level();
        frame.dispose();
        }
 else if(cmd.equals("How to Play")) {
```

```
                new HowtoPlay();
                frame.dispose();
            }
            else if(cmd.equals("Exit")) {
                System.exit(0);
                }
            }

           public static void main(String[] args) {
//          HowtoPlay gui = new HowtoPlay();
//          gui.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
//          gui.setVisible(true);
//          gui.pack();
//          gui.setTitle("How to Play!!!");

            new Main().setName();
           }
}
```

```
import java.util.*;


class Logic {

                int[][] blockS = {{ 4, 3, 5, 8, 7, 6, 1, 2, 9 }, { 8, 7, 6, 2, 1, 9, 3, 4, 5 },

                        { 2, 1, 9, 4, 3, 5, 7, 8, 6 }, { 5, 2, 3, 6, 4, 7, 8, 9, 1 }, { 9, 8, 1, 5, 2, 3, 4, 6, 7
},

                        { 6, 4, 7, 9, 8, 1, 2, 5, 3 }, { 7, 5, 4, 1, 6, 8, 9, 3, 2 }, { 3, 9, 2, 7, 5, 4, 6, 1, 8
},

                        { 1, 6, 8, 3, 9, 2, 5, 7, 4 }};

                //set number 1-9 for each box


                Random R_num = new Random();

                        // random numbers to exchange rows

                Random Grid_R_num = new Random();

                Random R_exnum = new Random();

                Random H_Rnum = new Random();


                int firstrow, secondrow, firstcol, secondcol, firstgrid, secondgrid, gc = 0;

                int carry[] = new int[9];

                int blockh[][] = new int[9][9];
```

```
int blockc[][] = new int[9][9];



int[][] generate() {

        //switching the rows


 int x = 10 + R_num.nextInt(10);
 for (int y = 0; y < x; y++) {
        for (int a = 0; a < 3; a++) {


            if (a == 0) {
                firstrow = R_num.nextInt(3);
                secondrow = R_num.nextInt(3);
            }


            else if (a == 1) {
                firstrow = 3 + R_num.nextInt(3);
                secondrow = 3 + R_num.nextInt(3);
            }
```

```
                else if (a == 2) {
                        firstrow = 6 + R_num.nextInt(3);
                        secondrow = 6 + R_num.nextInt(3);
                }

                for (int i = 0; i < 9; i++) {
                        carry[i] = blockS[firstrow][i];
                        blockS[firstrow][i] = blockS[secondrow][i];
                        blockS[secondrow][i] = carry[i];
                }
        }

                // switching the rows complete

        for (int a = 0; a < 3; a++) {
// switching the column
                if (a == 0) {
                        firstcol = R_num.nextInt(3);
                        secondcol = R_num.nextInt(3);
```

```
                }


                else if (a == 1) {
                        firstcol = 3 + R_num.nextInt(3);
                        secondcol = 3 + R_num.nextInt(3);
                }


                else if (a == 2) {
                        firstcol = 6 + R_num.nextInt(3);
                        secondcol = 6 + R_num.nextInt(3);
                }


                for (int i = 0; i < 9; i++) {
                        carry[i] = blockS[i][firstcol];
                        blockS[i][firstcol] = blockS[i][secondcol];
                        blockS[i][secondcol] = carry[i];
                }
        }
}
```

```
                    // Switchicng the column complete



                    // Switchicng the grids

firstgrid = 1 + Grid_R_num.nextInt(3);

secondgrid = 1 + Grid_R_num.nextInt(3);



if ((firstgrid == 1 && secondgrid == 2) || (firstgrid == 2 && secondgrid == 1)) {

        for (int i = 0; i < 3; i++)

                for (int j = 0; j < 9; j++) {

                        carry[j] = blockS[i][j];

                        blockS[i][j] = blockS[i + 3][j];

                        blockS[i + 3][j] = carry[j];

                }

} else if ((firstgrid == 2 && secondgrid == 3) || (firstgrid == 3 && secondgrid ==
2)) {

        for (int i = 3; i < 6; i++)

                for (int j = 0; j < 9; j++) {
```

```
                                        carry[j] = blockS[i][j];

                                        blockS[i][j] = blockS[i + 3][j];

                                        blockS[i + 3][j] = carry[j];

                                }

                        } else if ((firstgrid == 1 && secondgrid == 3) || (firstgrid == 3 && secondgrid ==
1)) {

                                for (int i = 0; i < 3; i++)

                                        for (int j = 0; j < 9; j++) {

                                                carry[j] = blockS[i][j];

                                                blockS[i][j] = blockS[i + 6][j];

                                                blockS[i + 6][j] = carry[j];

                                        }

                        }



                                                                        // Switching complete of tow
grids


                        int firstnum, secondnum, shuffle;


                                // suffling the puzzle
```

```
shuffle = 3 + R_num.nextInt(6);


for (int k = 0; k < shuffle; k++) {
        firstnum = 1 + R_exnum.nextInt(9);
        secondnum = 1 + R_exnum.nextInt(9);


        for (int i = 0; i < 9; i++)
                for (int j = 0; j < 9; j++) {
                        if (blockS[i][j] == firstnum) {
                                blockS[i][j] = secondnum;
                                continue;
                        }


                        if (blockS[i][j] == secondnum) blockS[i][j] = firstnum;
                }
}
return blockS;
}
```

```
int[][] save() {

                                        // will save the complete puzzle

 if (gc == 0) blockc = generate();


 gc = 1;


 return blockc;

}


 int[][] hide() {

                                // will hide number of puzzle

        for (int i = 0; i < 9; i++)

                for (int j = 0; j < 9; j++)

                        blockh[i][j] = blockc[i][j];


        int row, column, hidingnum;


        hidingnum =  50 + R_num.nextInt(10);


        for (int i = 0; i < hidingnum; i++) {
```

```
                    row = H_Rnum.nextInt(9);

                    column = H_Rnum.nextInt(9);



                    blockh[row][column] = 0;

                }

                return blockh;

            }

}
```

```
import java.util.Random;


class LogicHard {

                int[][] blockS = {{13, 1, 4, 14, 11, 16, 10, 9, 3, 6, 2, 12, 7, 8, 5, 15},

                        {16, 9, 10, 6, 5, 7, 12, 13, 14, 15, 8, 4, 11, 2, 3, 1},

                        {15, 12, 11, 2, 6, 3, 4, 8, 16, 7, 1, 5, 13, 14, 9, 10},

                        {3, 7, 8, 5, 1, 2, 14, 15, 10, 9, 11, 13, 4, 6, 12, 16},

                        {2, 14, 6, 1, 9, 5, 7, 4, 15, 13, 3, 16, 10, 11, 12, 8},

                        {9, 4, 7, 15, 10, 11, 1, 3, 8, 14, 6, 12, 13, 16, 5, 2},

                        {8, 13, 5, 16, 14, 2, 6, 12, 7, 4, 10, 11, 9, 1, 3, 15},

                        {12, 11, 3, 10, 16, 15, 13, 8, 2, 5, 9, 1, 7, 4, 6, 14},

                        {1, 7, 9, 10, 8, 2, 14, 13, 4, 15, 16, 6, 5, 12, 11, 3},

                        {15, 5, 11, 14, 1, 12, 9, 7, 2, 3, 13, 8, 6, 10, 4, 16},

                        {4, 8, 16, 3, 10, 5, 15, 6, 1, 11, 12, 7, 2, 9, 13, 14},

                        {6, 13, 2, 12, 11, 3, 16, 4, 14, 10, 5, 9, 8, 1, 15, 7},

                        {14, 3, 13, 7, 16, 9, 8, 5, 6, 10, 15, 11, 12, 4, 1, 2},

                        {4, 6, 16, 10, 12, 1, 15, 11, 7, 13, 2, 5, 3, 8, 14, 9},

                        {11, 15, 8, 9, 3, 10, 2, 4, 12, 16, 14, 1, 5, 6, 7, 13},
```

```
        {5, 12, 1, 2, 13, 14, 7, 6, 9, 8, 4, 3, 15, 16, 10, 11}};

        //set number 1-9 for each box

Random R_num = new Random();

        // random numbers to exchange rows
Random Grid_R_num = new Random();
Random R_exnum = new Random();
Random H_Rnum = new Random();

int firstrow, secondrow, firstcol, secondcol, firstgrid, secondgrid, gc = 0;
int carry[] = new int[16];
int blockh[][] = new int[16][16];
int blockc[][] = new int[16][16];


int[][] generate() {

        //switching the rows

 int x = 17 + R_num.nextInt(17);
```

```
for (int y = 0; y < x; y++) {

        for (int a = 0; a < 4; a++) {


                if (a == 0) {

                        firstrow = R_num.nextInt(4);

                        secondrow = R_num.nextInt(4);

                }


                else if (a == 1) {

                        firstrow = 4 + R_num.nextInt(4);

                        secondrow = 4 + R_num.nextInt(4);

                }


                else if (a == 2) {

                        firstrow = 8 + R_num.nextInt(4);

                        secondrow = 8 + R_num.nextInt(4);

                }

                else if (a == 3) {

                        firstrow = 12 + R_num.nextInt(4);

                        secondrow = 12 + R_num.nextInt(4);
```

```
                }


                for (int i = 0; i < 16; i++) {
                        carry[i] = blockS[firstrow][i];
                        blockS[firstrow][i] = blockS[secondrow][i];
                        blockS[secondrow][i] = carry[i];
                }
        }


                // switching the rows complete


        for (int a = 0; a < 4; a++) {

// switching the column


                if (a == 0) {
                        firstcol = R_num.nextInt(4);
                        secondcol = R_num.nextInt(4);
                }


                else if (a == 1) {
```

```
                    firstcol = 4 + R_num.nextInt(4);

                    secondcol = 4 + R_num.nextInt(4);

                }


                else if (a == 2) {

                    firstcol = 8 + R_num.nextInt(4);

                    secondcol = 8 + R_num.nextInt(4);

                }

                else if (a == 3) {

                    firstcol = 12 + R_num.nextInt(4);

                    secondcol = 12 + R_num.nextInt(4);

                }


                for (int i = 0; i < 16; i++) {

                    carry[i] = blockS[i][firstcol];

                    blockS[i][firstcol] = blockS[i][secondcol];

                    blockS[i][secondcol] = carry[i];

                }

            }

        }
```

```
                    // Switchicng the column complete



                    // Switching the grids
            firstgrid = 1 + Grid_R_num.nextInt(4);
            secondgrid = 1 + Grid_R_num.nextInt(4);



            if ((firstgrid == 1 && secondgrid == 2) || (firstgrid == 2 && secondgrid == 1)) {
                for (int i = 0; i < 4; i++)
                    for (int j = 0; j < 16; j++) {
                        carry[j] = blockS[i][j];
                        blockS[i][j] = blockS[i + 4][j];
                        blockS[i + 4][j] = carry[j];
                    }
            } else if ((firstgrid == 2 && secondgrid == 3) || (firstgrid == 3 && secondgrid ==
2)) {
                for (int i = 4; i < 8; i++)
                    for (int j = 0; j < 16; j++) {
```

```
                    carry[j] = blockS[i][j];
                    blockS[i][j] = blockS[i + 4][j];
                    blockS[i + 4][j] = carry[j];
                }
            } else if ((firstgrid == 3 && secondgrid == 4) || (firstgrid == 4 && secondgrid ==
3)) {
                for (int i = 8; i < 12; i++)
                    for (int j = 0; j < 16; j++) {
                        carry[j] = blockS[i][j];
                        blockS[i][j] = blockS[i + 4][j];
                        blockS[i + 4][j] = carry[j];
                    }
            } else if ((firstgrid == 1 && secondgrid == 3) || (firstgrid == 3 && secondgrid ==
1)) {
                for (int i = 0; i < 4; i++)
                    for (int j = 0; j < 16; j++) {
                        carry[j] = blockS[i][j];
                        blockS[i][j] = blockS[i + 8][j];
                        blockS[i + 8][j] = carry[j];
                    }
```

```
} else if ((firstgrid == 2 && secondgrid == 4) || (firstgrid == 4 && secondgrid ==
2)) {

        for (int i = 4; i < 8; i++)

            for (int j = 0; j < 16; j++) {

                carry[j] = blockS[i][j];

                blockS[i][j] = blockS[i + 8][j];

                blockS[i + 8][j] = carry[j];

            }

}



                                                    // Switching complete of tow
grids


int firstnum, secondnum, shuffle;

        // suffling the puzzle


shuffle = 4 + R_num.nextInt(8);


for (int k = 0; k < shuffle; k++) {

        firstnum = 1 + R_exnum.nextInt(16);
```

```
            secondnum = 1 + R_exnum.nextInt(16);


            for (int i = 0; i < 16; i++)
                    for (int j = 0; j < 16; j++) {
                            if (blockS[i][j] == firstnum) {
                                    blockS[i][j] = secondnum;
                                    continue;
                            }

                            if (blockS[i][j] == secondnum) blockS[i][j] = firstnum;
                    }
 }
 return blockS;
}


int[][] save() {

                                    // will save the complete puzzle
 if (gc == 0) blockc = generate();


 gc = 1;
```

```
 return blockc;

}


 int[][] hide() {

                                // will hide number of puzzle

            for (int i = 0; i < 16; i++)

                        for (int j = 0; j < 16; j++)

                                    blockh[i][j] = blockc[i][j];


            int row, column, hidingnum;


            hidingnum = 158 + R_num.nextInt(17);


            for (int i = 0; i < hidingnum; i++) {

                        row = H_Rnum.nextInt(16);

                        column = H_Rnum.nextInt(16);


                        blockh[row][column] = 0;

            }
```

```
                return blockh;

            }

}
```

```
import java.awt.*;

import java.awt.event.*;

import javax.swing.*;


public class Level implements ActionListener {


                        private JFrame frame;

                        private JPanel main, panel, btPanel, left, right, bottom;

                        private JButton easy, hard, back;

                        private JLabel name;


                        public Level() {

                         frame = new JFrame("Sudoku! by OOP-Project");

                         main = new JPanel(new BorderLayout());

                         panel = new JPanel(new GridLayout(2, 1));

                         btPanel = new JPanel(new GridLayout(3, 1, 10, 10));

                         left = new JPanel();

                         right = new JPanel();

                         bottom = new JPanel();

                         easy = new JButton("Easy");
```

```
hard = new JButton("Hard");

back = new JButton("Back");

name = new JLabel("Select Level");


panel.setBackground(Color.pink);

btPanel.setBackground(Color.pink);

main.setBackground(Color.pink);

left.setBackground(Color.pink);

right.setBackground(Color.pink);

bottom.setBackground(Color.pink);


name.setFont(new Font("ARIALBD", Font.ITALIC | Font.BOLD, 48));

name.setForeground(Color.black);

name.setHorizontalAlignment(SwingConstants.CENTER);


easy.setFont(new Font("ARIALBD", Font.BOLD, 20));

easy.setForeground(Color.black);

easy.setBackground(Color.green);

hard.setFont(new Font("ARIALBD", Font.BOLD, 20));

hard.setForeground(Color.black);
```

```
hard.setBackground(Color.red);

back.setForeground(Color.red);

back.setBackground(Color.black);

back.setFont(new Font("ARIALBD", Font.BOLD, 20));


easy.addActionListener(this);

hard.addActionListener(this);

back.addActionListener(this);


btPanel.add(easy);

btPanel.add(hard);


panel.add(name);

panel.add(btPanel);


left.add(back);


left.setPreferredSize(new Dimension(200, 480));

right.setPreferredSize(new Dimension(200, 480));

bottom.setPreferredSize(new Dimension(854, 50));
```

```
 main.add(panel, BorderLayout.CENTER);
 main.add(left, BorderLayout.WEST);
 main.add(right, BorderLayout.EAST);
 main.add(bottom, BorderLayout.SOUTH);

 frame.add(main);
 frame.setSize(854, 480);
 frame.setLocationRelativeTo(null);
 frame.setVisible(true);
}

public void actionPerformed(ActionEvent ev) {
 String cmd = ev.getActionCommand();
 if(cmd.equals("Easy")) {
            new Easy();
            frame.dispose();
            }
 else if (cmd.equals("Hard")){
            new Hard();
```

```
            frame.dispose();
        }
        else if(cmd.equals("Back")) {
            new Main();
            frame.dispose();
        }
        }
    }
```

```
public class Easy{

                        public Easy() {

                         GraphicallyRepresentation ob = new GraphicallyRepresentation();

                         ob.setTimer(new Timer());

                        }

}
```

```
public class Hard {

                        public Hard(){

                         GraphicallyRepresentationHard ob = new GraphicallyRepresentationHard();

                         ob.setTimer(new Timer());

                        }



}
```

```
import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;


import javax.swing.*;


public class HowtoPlay extends JFrame{

                        private ImageIcon image1, image2;

                        private JLabel label1;

                        private JButton back;


                        HowtoPlay(){

                         setTitle("How to play");


                         setLayout(new BorderLayout());


                         image1 = new ImageIcon(getClass().getResource("how2.png"));

                         label1 = new JLabel(image1);

                         add(label1, BorderLayout.CENTER);
```

```
back = new JButton("Back");
back.setFont(new Font("ARIALBD", Font.BOLD, 20));

back.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent arg0) {
                new Main();
                dispose();

        }
});

add(back, BorderLayout.SOUTH);

pack();
setVisible(true);
setLocationRelativeTo(null);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
```

```
import java.awt.*;

import java.awt.event.*;

import javax.swing.*;



class GraphicallyRepresentation extends JFrame implements ActionListener {

                        Container con;

                        JButton b[][] = new JButton[9][9];

                        TextField t[] = new TextField[61];

                        JMenuBar mbar;

                        JMenu file;

                        JMenuItem submit, exit;

                        Timer timer;



                        int[][] cp = new int[9][9];

                        int[][] ip = new int[9][9];



                        GraphicallyRepresentation() {

                         super("Sudoku! by OOP-Project");
```

```
setSize(854, 480);

// setresizeable(false);



con = getContentPane();

con.setLayout(new GridLayout(9, 9));



Mylogic ob1 = new Mylogic();

ob1.complete_puzzle();

ob1.puzzle();



int c = 0;

for (int i = 0; i < 9; i++)

        for (int j = 0; j < 9; j++) {

                b[i][j] = new JButton("" + ip[i][j]);

                b[i][j].setFont(new Font("ARIALBD", Font.BOLD, 20));

                b[i][j].setForeground(Color.red);
//ramdom number color

                if (ip[i][j] == 0) {

                        // b[i][j]=new JButton("");

                        b[i][j].setText("");

                        b[i][j].setBackground(Color.white);
```

```
                b[i][j].addActionListener(this);

            }

            con.add(b[i][j]);

            if (i == 3 || i == 4 || i == 5 || j == 3 || j == 4 || j == 5) {
                if (2 < i && i < 6 && 2 < j && j < 6) {
                    b[i][j].setBackground(Color.white);
//center square
                    continue;
                }
                b[i][j].setBackground(Color.black);
//2 4 6 8 square

            }

            else
                b[i][j].setBackground(Color.white);
// 1 3 7 9 color
        }
```

```
mbar = new JMenuBar();
                    //set function bar
setJMenuBar(mbar);


file = new JMenu("File");
                    //set name function


submit = new JMenuItem("Submit");
exit = new JMenuItem("Exit");


submit.addActionListener(new ActionListener() {
          public void actionPerformed(ActionEvent e) {
                    timer.stop();
                    int r = 0;
                    for (int i = 0; i < 9; i++)
                              for (int j = 0; j < 9; j++)
                                        if (cp[i][j] != Integer.parseInt(b[i][j].getText())) {
                                                  r = 1;
                                                  break;
                                        }
```

```
                    for (int i = 0; i < 9; i++) {
                        System.out.println();
                        for (int j = 0; j < 9; j++) {
                            System.out.print(cp[i][j]);
                            System.out.print(Integer.parseInt(b[i][j].getText())
+ "  ");
                        }
                    }
                    System.out.print("\n" + r);
                    //result game
                    if (r == 0) {

                JOptionPane.showMessageDialog(GraphicallyRepresentation.this, "You won the
Game");
                        // System.out.println("You won the Game");
                        timer.close();
                        new Main().setName();
                        dispose();
                    }
                    else {
                        // System.out.println("You lose the Game");
```

```
                JOptionPane.showMessageDialog(GraphicallyRepresentation.this, "You lose the
Game");
                                        timer.close();
                                        new Main().setName();
                                        dispose();
                                }
                                }
                        }
                );
                exit.addActionListener(new ActionListener() {
                //set exit function
                        public void actionPerformed(ActionEvent e) {
                                System.exit(0);
                                }
                        }
                );

                //about.addActionListener(new ActionListener() {
                //set about function
                        //public void actionPerformed(ActionEvent e) {
                                //JOptionPane.showMessageDialog(GraphicallyRepresentation.this,
```

```
                        //"58070058 Thanyathon project of java language",
                        //"How to play", JOptionPane.PLAIN_MESSAGE);
                    //}
                //}
//);

file.add(submit);
file.addSeparator();
file.add(exit);
mbar.add(file);
//mbar.add(about);

show();

// ob1.complet_puzzle();

MyWindowAdapter mwa = new MyWindowAdapter();
addWindowListener(mwa);

}
```

```
public void setTimer(Timer timer) {
 this.timer = timer;
}

class Mylogic extends Logic {
 void complete_puzzle() {
        cp = save();

 }

 void puzzle() {
        ip = hide();

 }

}

class MyWindowAdapter extends WindowAdapter {
```

```
 public void windowClosing(WindowEvent e) {
        System.exit(0);
 }
}

public void actionPerformed(ActionEvent e) {
 for (int i = 0; i < 9; i++)
        for (int j = 0; j < 9; j++) {
            if (e.getSource() == b[i][j]) {
                String s = JOptionPane.showInputDialog(null, "enter your
number");                    //set pop up answer window
                if(s != null && s.length() >= 1) {
                    int c = Integer.parseInt(s);
                    if (0 < c && 10 > c) {
                        b[i][j].setText(s);
                        b[i][j].setFont(new Font("ARIALBD",
Font.BOLD, 25));
                        b[i][j].setForeground(Color.green);
                            //set answer color
                    }
```

```
                                        }
                                        break;
                                    }
                                }
                            }

                            void recall() {
                             GraphicallyRepresentation rs = new GraphicallyRepresentation();
                            }
}
```

```
import java.awt.*;

import java.awt.event.*;

import javax.swing.*;


class GraphicallyRepresentationHard extends JFrame implements ActionListener{

                    Container conhard;

                    JButton b[][] = new JButton[16][16];

                    TextField t[] = new TextField[256];

                    JMenuBar mbarhard;

                    JMenu file;

                    JMenuItem submit, exit;

                    Timer timer;


                    int [][] cp = new int[16][16];

                    int [][] ip = new int[16][16];


                    GraphicallyRepresentationHard() {

                     super("Sudoku! by OOP-Project");

                     setSize(854, 480);
```

```
conhard = getContentPane();

conhard.setLayout(new GridLayout(16, 16));


MylogicHard ob2 = new MylogicHard();

ob2.complete_puzzle_hard();

ob2.puzzle_hard();


int c = 0;

for(int i = 0; i < 16; i++)

        for(int j = 0; j < 16; j++) {

                b[i][j] = new JButton("" + ip[i][j]);

                b[i][j].setFont(new Font("ARIALBD", Font.BOLD, 14));

                b[i][j].setForeground(Color.red);

                if (ip[i][j] == 0) {

                        b[i][j].setText("");

                        b[i][j].setBackground(Color.white);

                        b[i][j].addActionListener(this);

                }


                conhard.add(b[i][j]);
```

```
if(i == 0 || i == 1 || i == 2 || i == 3) {
        if(j == 0 || j == 1 || j == 2 || j ==3) {
                b[i][j].setBackground(Color.white);
        } else if (j == 8 || j == 9 || j == 10 || j ==11) {
                b[i][j].setBackground(Color.white);
        } else {
                b[i][j].setBackground(Color.black);
        }
} else if(i == 4 || i == 5 || i == 6 || i == 7) {
        if(j == 0 || j == 1 || j == 2 || j ==3) {
                b[i][j].setBackground(Color.black);
        } else if (j == 8 || j == 9 || j == 10 || j ==11) {
                b[i][j].setBackground(Color.black);
        } else {
                b[i][j].setBackground(Color.white);
        }
} else if(i == 8 || i == 9 || i == 10 || i == 11) {
        if(j == 0 || j == 1 || j == 2 || j ==3) {
                b[i][j].setBackground(Color.white);
```

```
                } else if (j == 8 || j == 9 || j == 10 || j ==11) {
                        b[i][j].setBackground(Color.white);
                } else {
                        b[i][j].setBackground(Color.black);
                }
            } else if(i == 12 || i == 13 || i == 14 || i == 15) {
                if(j == 0 || j == 1 || j == 2 || j ==3) {
                        b[i][j].setBackground(Color.black);
                } else if (j == 8 || j == 9 || j == 10 || j ==11) {
                        b[i][j].setBackground(Color.black);
                } else {
                        b[i][j].setBackground(Color.white);
                }
            }
        }

mbarhard = new JMenuBar();
setJMenuBar(mbarhard);

file = new JMenu("File");
```

```
//about = new JMenuItem("About");

submit = new JMenuItem("Submit");

exit = new JMenuItem("Exit");


submit.addActionListener(new ActionListener() {

        public void actionPerformed(ActionEvent e) {

                timer.stop();

                int r = 0;


                for (int i = 0; i < 16; i++)

                        for (int j = 0; j < 16; j++)

                                if(cp[i][j] != Integer.parseInt(b[i][j].getText())) {

                                        r = 1;

                                        break;

                                }


                for (int i = 0; i < 16; i++) {

                        System.out.println();
```

```
                    for (int j = 0; j < 16; j++) {
                        System.out.print(cp[i][j]);
                        System.out.print(Integer.parseInt(b[i][j].getText())
+ "  ");
                    }
                }

                System.out.print("\n" + r);

                if (r == 0) {

                    JOptionPane.showMessageDialog(GraphicallyRepresentationHard.this, "You won
the Game");
                        timer.close();
                        new Main().setName();;
                        dispose();
                }
                else {

                    JOptionPane.showMessageDialog(GraphicallyRepresentationHard.this, "You lose
the Game");
                        timer.close();
```

```
                                        new Main().setName();;
                                        dispose();
                                        }
                                }
                        }
                );

                exit.addActionListener(new ActionListener(){
                        public void actionPerformed(ActionEvent e) {
                                System.exit(0);
                                }
                        }
                );

                //about.addActionListener(new ActionListener() {
                        //public void actionPerformed(ActionEvent e) {

                //JOptionPane.showMessageDialog(GraphicallyRepresentationHard.this, "58070058
Thanyathon project of java language", "How to play", JOptionPane.PLAIN_MESSAGE);
                                //}
                        //}
```

```
//);


file.add(submit);

file.addSeparator();

file.add(exit);

mbarhard.add(file);

//mbarhard.add(about);


show();


MyWindowAdapterHard mwa = new MyWindowAdapterHard();

addWindowListener(mwa);

}

public void setTimer(Timer timer) {

this.timer = timer;

}


class MylogicHard extends LogicHard{

void complete_puzzle_hard() {

        cp = save();
```

```
 }
 void puzzle_hard() {
        ip = hide();
 }
}

class MyWindowAdapterHard extends WindowAdapter{
 public void windowClosing(WindowEvent e) {
        System.exit(0);
 }
}

public void actionPerformed(ActionEvent e) {
 for(int i = 0; i < 16; i++)
        for(int j = 0; j < 16; j++) {
            if(e.getSource() == b[i][j]) {
                String s = JOptionPane.showInputDialog(null, "enter your
number");

                if(s != null && s.length() >= 1) {
                    int c1 = Integer.parseInt(s);
```

```
                                        if (0 < c1 && 17 > c1) {
                                            b[i][j].setText(s);
                                            b[i][j].setFont(new Font("ARIALBD",
Font.BOLD, 25));
                                            b[i][j].setForeground(Color.green);
                                                //set answer color
                                        }
                                    }

                                    break;
                                }
                            }
                        }

                        void recall() {
                         GraphicallyRepresentationHard rs = new GraphicallyRepresentationHard();
                        }
}
```

```
import java.awt.BorderLayout;

import java.awt.Font;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.FileWriter;

import java.io.IOException;


import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.SwingConstants;

import javax.swing.WindowConstants;


public class Timer extends JFrame{


                private JLabel time;

                private boolean stopFlag;

                private int sec, min, hour;


                public Timer() {
```

```
setLayout(new BorderLayout());
time = new JLabel("");
time.setText("----------------");
time.setHorizontalAlignment(SwingConstants.CENTER);
time.setFont(new Font("ARIALBD", Font.ITALIC | Font.BOLD, 36));
setDefaultCloseOperation(WindowConstants.DO_NOTHING_ON_CLOSE);

add(time, BorderLayout.CENTER);

setVisible(true);
pack();
new Thread(new Runnable() {

    @Override
    public void run() {
        while(!stopFlag) {
            try {
                sec++;
                if(sec == 60) {
                    sec = 0;
```

```
                    min++;
                }else if(min == 60) {
                    min = 0;
                    hour++;
                }
                String format =
String.format("%02d:%02d:%02d", hour, min, sec);
                time.setText(format);
                Thread.sleep(1000);

            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }

    }
 }).start();
}

public void stop() {
```

```
 stopFlag = true;
}

public void close() {
 try(FileWriter fw = new FileWriter("record.txt", true)){
        fw.write(Main.getName() + " " +  time.getText() + "\r\n");
 } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
 }
 dispose();
}

}
```