**JAVA程序设计**

**课程设计报告**

实验名称： 连连看小游戏

学 院： 信息技术学院

班 级： 14 计算机科学与技术

成 员： 肖 洁（084014136）

高桂春（084014143）

**信息技术学院**

**南京中医药大学**

目录

[一、课程设计的目的和内容 3](#_Toc471495179)

[1.1、目的 3](#_Toc471495180)

[1.2、内容 3](#_Toc471495181)

[二、系统需求分析 3](#_Toc471495182)

[2.1、系统目标 3](#_Toc471495183)

[2.2、主体功能 4](#_Toc471495184)

[2.3、开发环境 4](#_Toc471495185)

[三、系统概要设计 4](#_Toc471495186)

[3.1、系统功能模块划分 4](#_Toc471495187)

[3.1.1、开始游戏 5](#_Toc471495188)

[3.1.2、使用“炸弹” 5](#_Toc471495189)

[3.1.3、使用“刷新” 5](#_Toc471495190)

[3.1.4、对用户游戏时的判断 5](#_Toc471495191)

[3.2、系统流程图 5](#_Toc471495192)

[3.3、系统UML图 6](#_Toc471495193)

[四、系统详细设计 7](#_Toc471495194)

[4.1、设计思路 7](#_Toc471495195)

[4.2、设计方法 8](#_Toc471495196)

[4.3、算法设计 9](#_Toc471495197)

[五、实验结果 10](#_Toc471495198)

[5.1、菜单栏 10](#_Toc471495199)

[5.2、工具栏 12](#_Toc471495200)

[六、测试 15](#_Toc471495201)

[七、人员分工 15](#_Toc471495202)

[八、小结及设计体会 15](#_Toc471495203)

[附录 关键代码 17](#_Toc471495204)

# 课程设计的目的和内容

1.1、目的

为了进一步巩固Java程序设计课程所学到的知识，深刻把握Java语言的重要概念及其面向对象的特性，锻炼熟练的应用面向对象的思想和设计方法解决实际问题的能力，巩固所学Java语言基本知识，增进Java语言编辑基本功，掌握JDK、Eclipse等开发工具的运用，拓宽常用类库的应用，把所学课程及相关知识加以融会贯通，全面掌握Java语言的变成思想及面向对象程序设计的方法，为今后从事实际工作打下坚实的基础。

1.2、内容

连连看游戏是让用户每次选择两个图形，如果图形满足一定条件（两个图形一样，且这两个图形之间存在转弯少于3的路径），则两个图形都能消掉。给定任意具有相同图形的两个格子，我们需要寻找这两个格子之间在转弯最少的情况下，经过格子数目最少的路径。如果这个最优路径的转弯数目少于3 ，则这两个格子可以消去。在此条件下将连连看小游戏的功能进行丰富，使用户能得到更多更好的游戏体验。

# 系统需求分析

2.1、系统目标

连连看游戏的目标是让用户尽快在不同图案的图片之中快速选择出相同的图片，并且图片之间存在转弯少于3的路径，然后进行消除。游戏规则为当用户选择点后，如果选择的第二个点与第一个点之间无法消除，扣1分，如果有解，则增加10分；游戏限定时间为地图中方块数 \* 2，比如说有120个方块，就应该在240秒内完成，超过240秒没有时间奖励分。时间奖励分为剩余时间 \* 2，即每提前一秒钟奖励2分；允许用户使用刷新功能，系统会自动判断当前地图有无解，如果有解，扣8分，无解则加8分；同时用户可以使用提示功能，如果提示成功，扣10分；使用炸弹功能，如果炸弹功能有效，扣12分。最后对用户的总成绩进行记录。

2.2、主体功能

通过JAVA编写一个连连看程序，通过鼠标点击使相同的两个图像消除。程序实现以下几个功能：

1. 判断鼠标点击的两张图片是否相同；
2. 判断两张图片是否可以相连；
3. 判断是否点击刷新；
4. 判断是否点击炸弹；
5. 判断剩余图片，是否胜利。

2.3、开发环境

1、Windows7系统

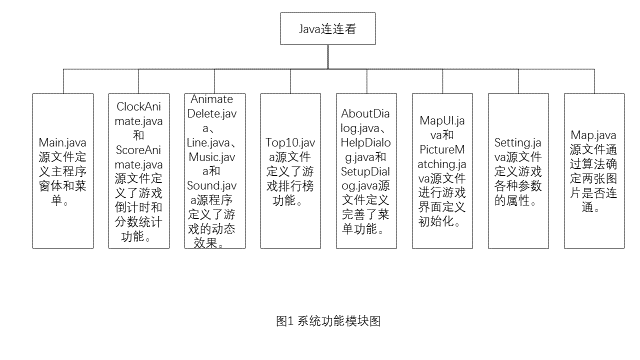
2、IntelliJ IDEA软件

3、JDK 1.8

# 三、系统概要设计

3.1、系统功能模块划分

此设计共有以下几个功能模块：



3.1.1、开始游戏

游戏开始后，程序会自动载入图片，并通过特定的算法排列图片，并标记图片，以便判断图片是否相同。记录图片坐标，以方便计算图片是否可以相连。确定图片大小，以确定用户点击的是哪张图片。

3.1.2、使用“炸弹”

用户点击菜单中的“炸弹”选项使用，点击后如果界面上有两张可以相连的相同图片就随机消除两张可以相连的相同的图片。

3.1.3、使用“刷新”

用户点击菜单中的“刷新”选项使用，点击后重置界面，让用户可以重新进行消除。

3.1.4、对用户游戏时的判断

用户选择两张一样的图片，通过鼠标进行选取，如果用户两次点击的图片不相同，那么系统认为用户没有进行点击（鼠标点击数记为0），以完成用户以后的选取工作。如果用户选择的是相同的图片，那么系统会消除这两张图片，并使鼠标点击数记为0,出这两张图片的标记，以便用户可以进行以后的选择。

3.2、系统流程图

由系统分析可得系统流程图如下：

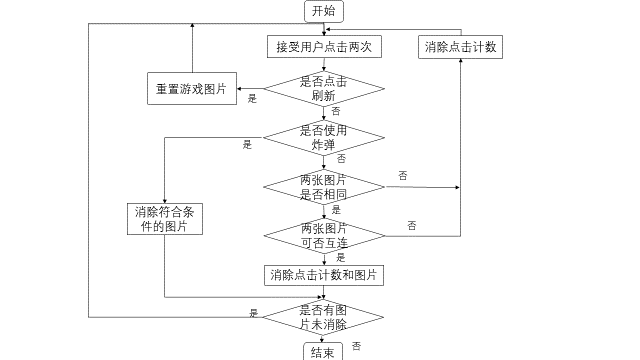


图2 系统流程图

3.3、系统UML图

由程序分析可得出UML图如下：

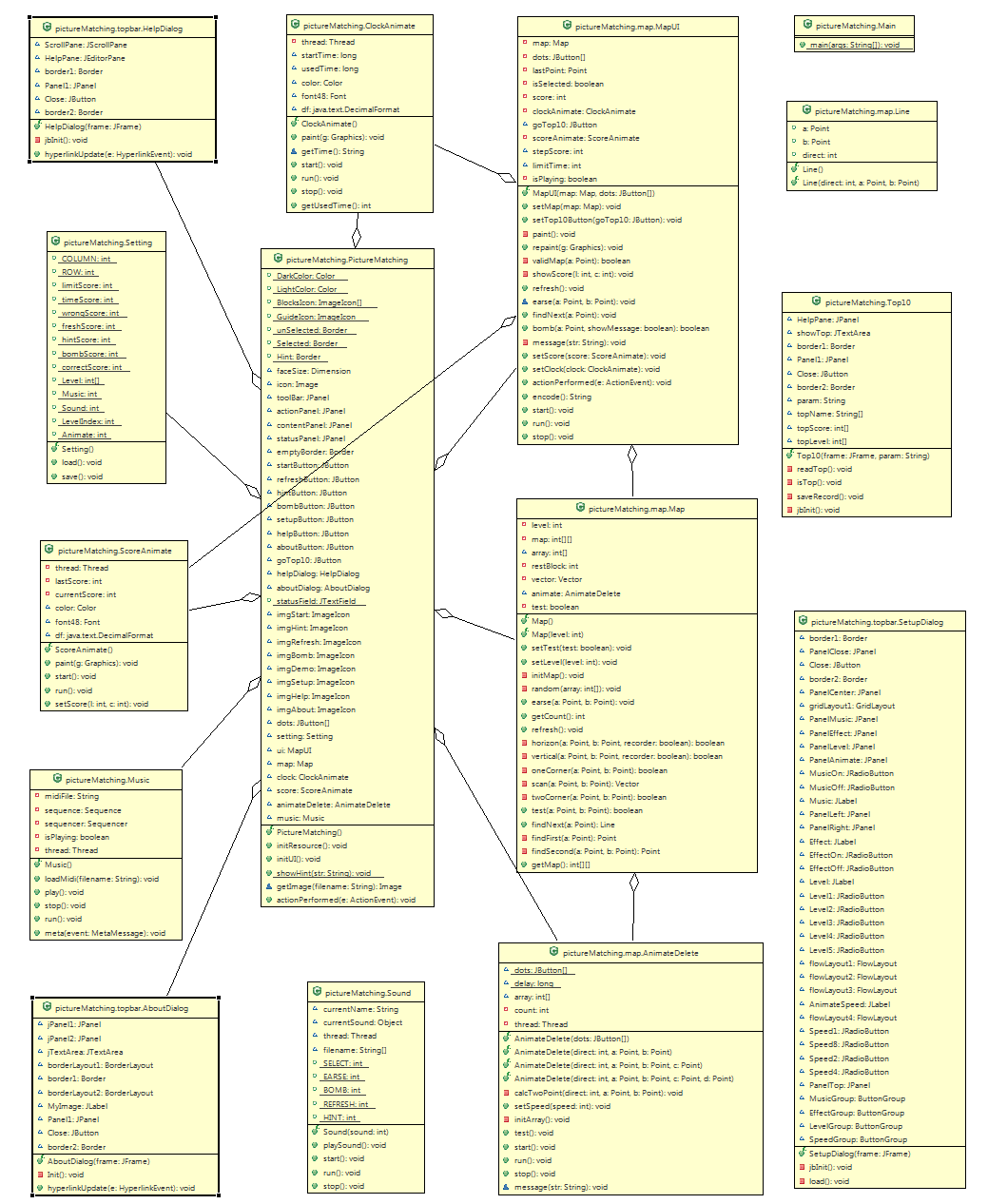


图3 系统UML图

# 四、系统详细设计

4.1、设计思路

1）用户可以通过点击菜单中的“开始”按钮开始新游戏程序。程序会载入图片，并通过特定的算法排列图片，并标记图片以判断图片是否相同。记录图片坐标，以方便计算图片是否可以相连。并确定用户点击的是哪一张图片。

2）用户选择两张一样的图片，通过鼠标进行选取，如果用户两次点击的图片不相同，那么系统认为用户没有进行点击（鼠标点击数记为0），以完成用户以后的选取工作。如果用户选择的是相同的图片，那么系统会消除这两张图片，并使鼠标点击数记为0,出这两张图片的标记，以便用户可以进行以后的选择。用户每成功消除一次积分增加10分。

3）用户也可以通过使用“炸弹”功能进行消除图片。方法是点击菜单中的“炸弹”选项使用，点击后如果界面上有两张可以相连的相同图片就随机消除两张可以相连的相同图片。用户每使用一次“炸弹”功能，积分减少12分。

4）用户可以使用“刷新”功能重置当前游戏界面。方法是点击菜单中的“刷新”选项使用，点击后界面中所有的图片都发生重置。用户进行刷新时若当前地图有解则扣除积分8分，无解则增加积分8分。

5）游戏限定时间为地图中方块数 \* 2，比如说有120个方块，就应该在240秒内完成，超过240秒没有时间奖励分，时间奖励分为剩余时间 \* 2，即每提前一秒钟奖励2分。

6）系统还提供音乐以及难度设置，用户可根据自己的需要进行修改。

4.2、设计方法

在设计连连看小游戏时，需要编写15个java源文件：Main.java，AnimateDelete.java，Line.java，Map.java，MapUI.java，AboutDialog.java，HelpDialog.java，SetupDialog.java，ClockAnimate.java，Music.java，PictureMatching.java，ScoreAnimate.java，Setting.java，Sound.java，Top10.java。

1. Main.java：主窗口部分定义了程序用函数，实现了程序的使用。
2. AnimateDelete.java：实现消除时的动画消除效果。
3. Line.java：实现游戏过程中的连线功能。
4. Map.java：通过算法确定两张图片是否连通。
5. MapUI.java：设计游戏界面，游戏部分判断点击的图片，判断两图片是否可连，是否相同，判断游戏是否胜利。实现炸弹功能和刷新功能。
6. AboutDialog.java：完成菜单栏中“关于”的功能。
7. HelpDialog.java：完成菜单栏中“帮助”的功能。
8. SetupDialog.java：完成菜单栏中“设置”的功能。
9. ClockAnimate.java：计算时间并对时间进行监听。

10）Music.java：进行音乐设置。

11）PictureMatching.java：对Swing界面进行初始化操作。

12）ScoreAnimate.java：对用户在游戏过程中的分数进行操作。

13）Setting.java：定义游戏中各种参数的属性。

14）Sound.java：进行音效设置。

15）Top10.java：设置游戏的排行榜。

4.3、算法设计

在检验两个方块能否消掉的时候，我们要让两个方块同时满足两个条件才行，就是两者配对并且连线成功。

\*分3种情况：(从下面的这三种情况，我们可以知道，需要三个检测，这三个检测分别检测一条直路经。这样就会有三条路经。若这三条路经上都是空按钮，那么就刚好是三种直线（两个转弯点）把两个按钮连接起来了)

\* 1.相邻

\* 2. 若不相邻的先在第一个按钮的同行找一个空按钮。1).找到后看第二个按钮横向到这个空按钮所在的列是否有按钮。2).没有的话再看第一个按钮到与它同行的那个空按钮之间是否有按钮。3).没有的话，再从与第一个按钮同行的那个空按钮竖向到与第二个按钮的同行看是否有按钮。没有的话路经就通了,可以消了.

\* 3.若2失败后，再在第一个按钮的同列找一个空按钮。1).找到后看第二个按钮竖向到这个空按钮所在的行是否有按钮 2).没有的话，再看第一个按钮到与它同列的那个空按钮之间是否有按钮。3).没有的话，再从与第一个按钮同列的那个空按钮横向到与第二个按钮同列看是否有按钮。没有的话路经就通了，可以消了。

\* 若以上三步都失败，说明这两个按钮不可以消去。

# 五、实验结果

5.1、菜单栏

1）点击设置，出现游戏设置相关内容：

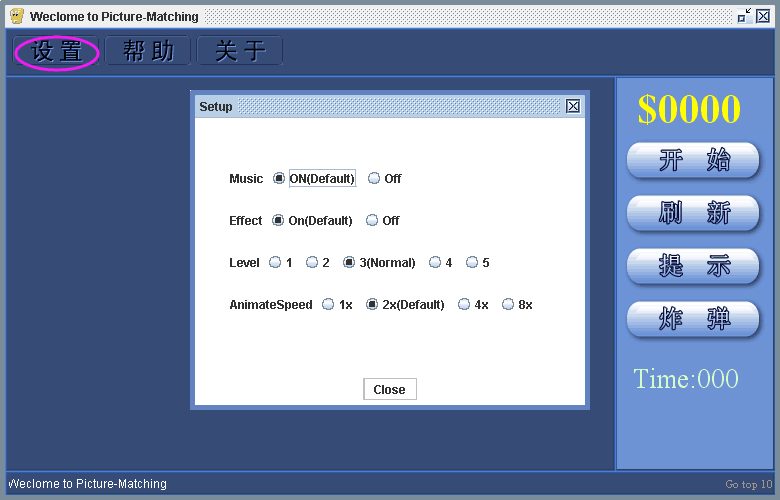


图5-1-1 设置界面

2）点击帮助，显示游戏帮助信息：

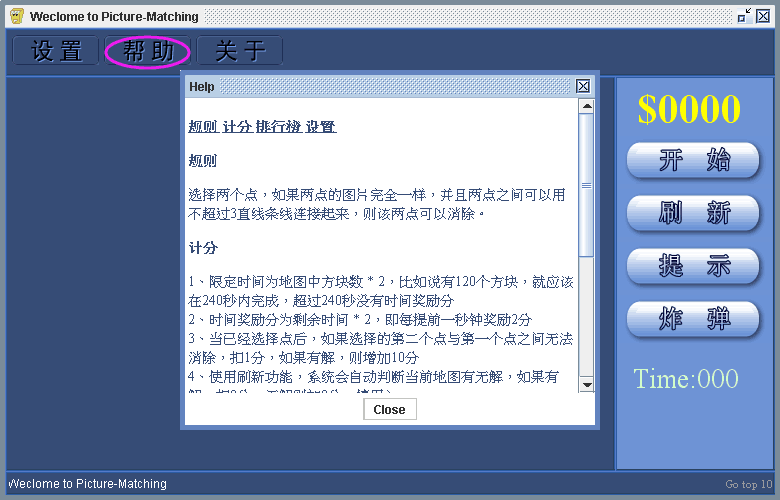


图5-1-2 帮助界面

3）点击关于，显示游戏开发信息：

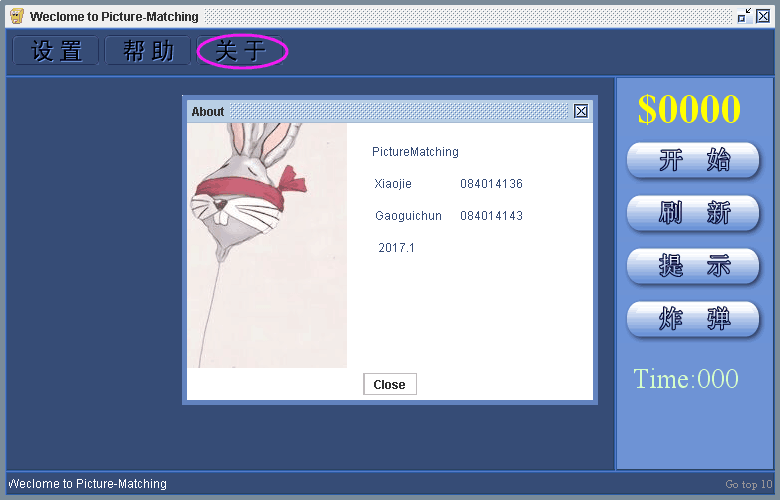


图5-1-3 关于界面

5.2、工具栏

1）点击开始，游戏开始，用户可以进行操作：



图5-2-1 开始界面

2）点击刷新，游戏布局重置：



图5-2-2 刷新界面

3）点击提示，系统自动给出可消除图片提示：



图5-2-3 提示界面

1. 点击炸弹，系统自动选择一组可消除图片消除：



图5-2-4 炸弹界面

5）游戏结束后，点击Go top 10，系统弹出输入框输入游戏名进行排名，若分数不在Top10内弹出如下信息框：

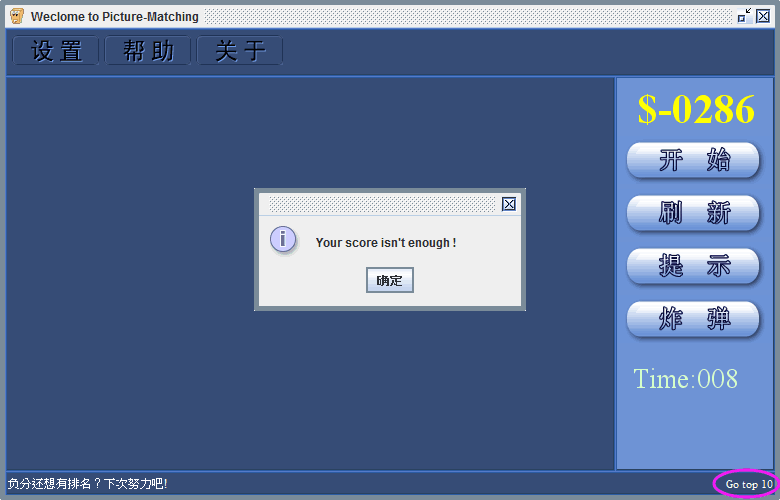


图5-2-6 排行榜排名提示界面

5）点击输入框/消息框的确定按钮,进入排行榜:

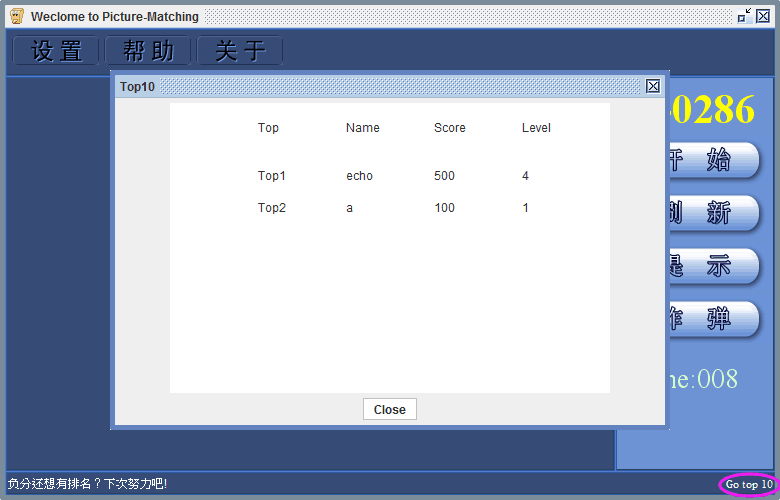


图5-2-7 排行榜界面

# 六、测试

本程序采用黑盒测试和单元测试两种测试方法。

在程序开发过程中，我们用单元测试对所写出的每一个类进行测试，来验证该段代码的功能是否能正确实现。可以为后续的开发提供支援，并且避免了代码出现回归。

程序开发结束后，我们采用黑盒测试的方法，在测试中，把程序看作一个不能打开的黑盒子，在完全不考虑程序内部结构和内部特性的情况下，在[程序接口](http://baike.so.com/doc/7844608-8118703.html)进行测试，它只检查程序功能是否按照需求规格说明书的规定正常使用，程序是否能适当地接收输入数据而产生正确的输出信息。试图发现下列几类错误。功能不正确或遗漏;界面错误;输入和输出错误;[数据库](http://baike.so.com/doc/5447468-5685836.html)访问错误;性能错误;[初始化](http://baike.so.com/doc/6380882-6594531.html)和[终止](http://baike.so.com/doc/6019883-6232879.html)错误等。

经过两轮测试，我们基本消除了程序中的bug，是程序能更完整安全的运行。

# 七、人员分工

本小组成员为肖洁（084014136）和高桂春（084014143）两人，在课程设计的过程中，程序代码部分和设计报告由两人合作完成。

# 八、小结及设计体会

这次课程设计真的是困难重重，一开始我们小组选择了“黄金矿工”作为我们设计的目标，并为此奋斗了很久，然而设计游戏的过程总是艰辛的，我们在没有任何帮助的情况下，只能完成到一半，就因为线程实现困难，当钩子下放以后屏幕刷新过快导致屏幕不停闪烁的问题无法解决而不得不放弃“黄金矿工”这个课题。在这之后，我们又花了一下午的时间，最终确定改换“连连看”为我们课程设计的题目，在这之后又开始新一轮的奋斗。前功尽弃的感受是很痛苦的，可是我们仍然要全力以赴去完成这一次的课设。

通过这次课程设计使我们懂得了理论与实际相结合是很重要的，只有理论知识是远远不够的，只有把所学的理论知识与实践相结合起来，从理论中得出结论，才能真正为社会服务，从而提高自己的实际动手能力和独立思考的能力。在设计的过程中遇到问题，可以说得是困难重重，并且还发现了自己的许多不足之处，对以前所学过的知识理解得不够深刻，掌握得不够牢固。但通过这次课程设计之后，我们一定会把以前所学过的知识重新温故。

在本次JAVA课程设计中，我们更加充分的理解了课本上的知识，并能够加以扩展，从而应用于实践当中，这几天的课程设计令我们受益匪浅，很多平时模棱两可的知识点都认真复习并实践了。我们对JAVA提升了认识，我们意识到我们所学的东西将来都是要付诸实践的，所以一切要从实际情况出发，理论联系实际，这样才能真正发挥我们所具备的能力。经过此次课程设计，我们向成功的目标又迈进了一步。

我们完成了这次JAVA课程设计，不过这只是我们学习路上的驿站。我们相信我们在未来的学习中会记住从本次课程设计中所学到的东西，并将它用到未来的学习中去。在这里谢谢老师的指导，我们以后会更加努力地学习。

# 附录 关键代码

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 类名: Map

\* 函数: initMap

\* 作用:“开始”算法

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void initMap() {

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

array[i \* 4] = i + 1;

array[i \* 4 + 1] = i + 1;

array[i \* 4 + 2] = i + 1;

array[i \* 4 + 3] = i + 1;

}

random(array);

for(int i = 0; i < 10; i++){

for(int j = 0; j < 17; j++)

map[i][j] = array[i \* 17 + j];

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 类名: Map

\* 函数: refresh

\* 作用:“刷新”算法

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void refresh() {

int count = getCount();

if(count <= 0) return;

int temp[] = new int[count];

count = 0;

for(int row = 0; row < 10; row++){

for(int col = 0; col < 17; col++)

if(map[row][col] > 0) {

temp[count] = map[row][col];

count++;

}

}

random(temp);

count = 0;

for(int row = 0; row < 10; row++) {

for(int col = 0; col < 17; col++)

if(map[row][col] > 0) {

map[row][col] = temp[count];

count++;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 类名: Map

\* 函数: findNext

\* 作用:“提示”算法

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public Line findNext(Point a) {

Point b = new Point();

a = findFirst(a);

if(a.equals(new Point(-1, -1))) return new Line(0, a, b);

for(; !a.equals(new Point(-1, -1)); a = findFirst(a))

for(b = findSecond(a, b); !b.equals(new Point(-1, -1)); b = findSecond(a, b))

if(test(a, b)) return new Line(1, a, b);

return new Line(0, a, b);

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 类名: PictureMatching

\* 函数: initUI

\* 作用: 初始化用户界面

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void initUI() {

//界面整体布局

Border border = BorderFactory.createBevelBorder(BevelBorder.LOWERED, new Color(45, 92, 162), new Color(43, 66, 97), new Color(45, 92, 162), new Color(84, 123, 200));

BorderLayout borderLayout = new BorderLayout();

toolBar.setBackground(DarkColor);

toolBar.setBorder(border);

toolBar.setPreferredSize(new Dimension(780, 48));

toolBar.setMinimumSize(new Dimension(780, 48));

toolBar.setLayout(new FlowLayout(FlowLayout.LEFT));

actionPanel.setBackground(LightColor);

actionPanel.setBorder(border);

actionPanel.setPreferredSize(new Dimension(160, 380));

actionPanel.setMinimumSize(new Dimension(160, 380));

contentPanel.setBackground(DarkColor);

contentPanel.setBorder(border);

contentPanel.setPreferredSize(new Dimension(620, 380));

contentPanel.setMinimumSize(new Dimension(620, 380));

statusPanel.setBackground(DarkColor);

statusPanel.setBorder(border);

statusPanel.setPreferredSize(new Dimension(620, 24));

statusPanel.setMinimumSize(new Dimension(620, 24));

statusPanel.setLayout(new BorderLayout());

this.getContentPane().setLayout(borderLayout);

this.getContentPane().add(toolBar, BorderLayout.NORTH);

this.getContentPane().add(actionPanel, BorderLayout.EAST);

this.getContentPane().add(contentPanel, BorderLayout.CENTER);

this.getContentPane().add(statusPanel, BorderLayout.SOUTH);

//加入地图

contentPanel.add(ui);

//加入计分

actionPanel.add(score);

//加入开始按钮

actionPanel.add(startButton);

startButton.setBorder(emptyBorder);

startButton.setIcon(imgStart);

startButton.addActionListener(this);

//加入刷新按钮

actionPanel.add(refreshButton);

refreshButton.setBorder(emptyBorder);

refreshButton.setIcon(imgRefresh);

refreshButton.addActionListener(this);

//加入提示按钮

actionPanel.add(hintButton);

hintButton.setBorder(emptyBorder);

hintButton.setIcon(imgHint);

hintButton.addActionListener(this);

//加入炸弹按钮

actionPanel.add(bombButton);

bombButton.setBorder(emptyBorder);

bombButton.setIcon(imgBomb);

bombButton.addActionListener(this);

//加入设置

toolBar.add(setupButton);

setupButton.setBorder(emptyBorder);

setupButton.setIcon(imgSetup);

setupButton.addActionListener(this);

//加入帮助

toolBar.add(helpButton);

helpButton.setBorder(emptyBorder);

helpButton.setIcon(imgHelp);

helpButton.addActionListener(this);

//加入关于

toolBar.add(aboutButton);

aboutButton.setBorder(emptyBorder);

aboutButton.setIcon(imgAbout);

aboutButton.addActionListener(this);

//加入时钟

actionPanel.add(clock);

//加入状态栏

statusPanel.add(statusField, BorderLayout.CENTER);

statusField.setBorder(unSelected);

statusField.setEditable(false);

statusField.setForeground(Color.white);

statusField.setBackground(DarkColor);

//加入发送按钮

statusPanel.add(goTop10, BorderLayout.EAST);

goTop10.setBorder(BorderFactory.createEmptyBorder(0, 0, 0, 0));

goTop10.setForeground(Color.white);

goTop10.setBackground(DarkColor);

goTop10.setFont(new Font("Serif", Font.PLAIN, 11));

goTop10.addActionListener(this);

goTop10.setEnabled(false);

}