**连连看小游戏实验报告**

# 实验目的

### 了解如何将程序文件夹导入Linux系统中，并在Linux系统环境下正确运行，熟悉Linux的操作环境。

### 利用Java编程语言实现小游戏界面设置和连线消除的游戏效果。

# 实验环境

Linux操作系统或虚拟机；

Java配置：Jdk 1.8；

Java开发工具：eclipse；

# 代码实现功能

### 实现一个小型的连连看游戏界面；

### 开始游戏后，在8\*8的地图上随机生成图片，位置随机打乱；

### 设置时间沙漏，规定在一定时间内将相同的图片连线，连线完成后选择进入下一关或是结束游戏；

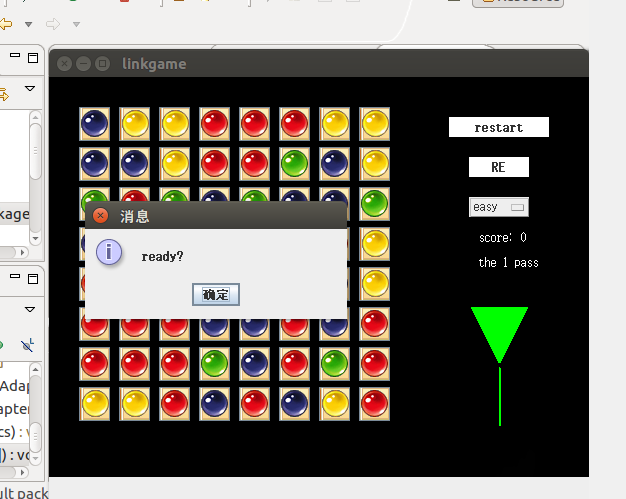
### 连线时要保证呈直线连接且不能穿过任何一个中间图片，否则无效。消除成功后画面中的图片消失，继续游戏至游戏结束或是时间用完；

### 游戏过程中，遇到困难玩家可以点击RE（刷新键）重新打乱地图上的图片；

### 如果有些玩家觉得游戏难度太低，可以选择游戏的难易程度。

# 实验内容

## 游戏界面



**部分代码：**

**public** **void** newMap() {//生成新的游戏界面

ArrayList<Integer> numbers = **new** ArrayList<Integer>();

**for** (**int** i = 0; i < Kinds; i++) {

numbers.add(i + 1);//加到列表尾部

numbers.add(i + 1);

}//每一次重新布局的时候，能保证一定有前几种难度中的图片类型

Random random = **new** Random();

**int** temp = 0;

**for** (**int** i = 0; i < 32- Kinds; i++) {

temp = random.nextInt(Kinds) + 1;//0~kinds-1之间的随机数在加1

numbers.add(temp);

numbers.add(temp);

}

Collections.*shuffle*(numbers);//随机打乱原来的顺序

map = **new** **int**[8][8];

temp = 0;

**for** (**int** i = 0; i < 8; i++) {

**for** (**int** j = 0; j < 8; j++) {

//JOptionPane.showMessageDialog(null, numbers.get(temp));

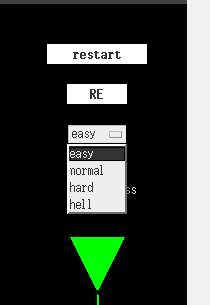
map[i][j] = numbers.get(temp++).intValue();//get方法返回第i个元素，intvalue 返回int类型

}

}

}

## 难易程度选择（简单、一般、困难、变态难）



**部分代码：**

**private** **static** **final** **long** ***serialVersionUID*** = 1L;//序列化时为了保持版本的兼容性

**private** **int**[][] map = **new** **int**[8][8];//8\*8的正方形

**private** **int** kind, randomx, randomy, randomx1, randomy1; // 种类，随机x

**private** **int** coordinatex, coordinatey, coordinatex1, coordinatey1; // 坐标X

**private** Point lineStart = **new** Point(0, 0);

**private** **int** clicktimes;

**private** **int** jishushengyu;//计数剩余

**private** **int** Kinds = 4;

**private** **int** score;

**private** **int** guanshu;//关数

loudou ld = **new** loudou();// 漏斗

JButton BlockButton[][] = **new** JButton[8][8];//

Choice difficultChoice = **new** Choice();

JButton newgameButton = **new** JButton("restart");

JButton reLoad = **new** JButton("RE");

ImageIcon ii = **new** ImageIcon("src/im/bk.jpg");

ImageIcon aIcon = **new** ImageIcon("src/im/1.gif");

ImageIcon bIcon = **new** ImageIcon("src/im/2.gif");

ImageIcon cIcon = **new** ImageIcon("src/im/3.gif");

ImageIcon dIcon = **new** ImageIcon("src/im/4.gif");

ImageIcon eIcon = **new** ImageIcon("src/im/5.gif");

ImageIcon fIcon = **new** ImageIcon("src/im/6.gif");

ImageIcon gIcon = **new** ImageIcon("src/im/7.gif");

ImageIcon hIcon = **new** ImageIcon("src/im/8.gif");

ImageIcon iIcon = **new** ImageIcon("src/im/9.gif");

ImageIcon jIcon = **new** ImageIcon("src/im/10.gif");

ImageIcon kIcon = **new** ImageIcon("src/im/11.gif");

ImageIcon lIcon = **new** ImageIcon("src/im/12.gif");

ImageIcon mIcon = **new** ImageIcon("src/im/13.gif");

ImageIcon nIcon = **new** ImageIcon("src/im/14.gif");

ImageIcon oIcon = **new** ImageIcon("src/im/15.gif");

**public** **void** itemStateChanged(ItemEvent e) {

// **TODO** 自动生成的方法存根

**if** (e.getSource() == difficultChoice) {

String selected = difficultChoice.getSelectedItem();

**if** (selected == "easy") {

Kinds = 4;

newGame();

repaint();

} **else** **if** (selected == "normal") {

Kinds = 8;

newGame();

repaint();

} **else** **if** (selected == "hard") {

Kinds = 12;

newGame();

repaint();

} **else** **if** (selected == "hell") {

Kinds = 15;

newGame();

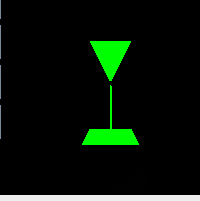
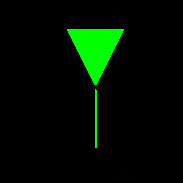
repaint();

}

}

}

## 计时漏斗实现（时间流逝，沙漏会向下漏）



部分代码：

**class** loudou **extends** JPanel **implements** Runnable {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** **int** dijiguan;

**int** remainTimes = 0; // 时间

**int** x1 = 0;

**int** y1 = 30;

**int** x2 = 60;

**int** y2 = 150;

Thread nThread1;//线程

JLabel overJLabel = **new** JLabel();

JDialog dialog = **new** JDialog();

**public** loudou() {

nThread1 = **new** Thread(**this**);

nThread1.start();

**this**.setLayout(**null**);

**this**.add(overJLabel);

overJLabel.setBounds(0, 0, 200, 50);

overJLabel.setForeground(Color.***white***);

}

**public** **void** setdijiguan(**int** x) {

**this**.dijiguan = x;

}

**public** **void** paintComponent(Graphics g) // 画画函数

{

**super**.paintComponent(g);

g.setColor(Color.***green***);

**for** (**int** i = 0; i < 56; i++) {

g.drawLine(x1 + i / 2 + 2, y1 + i, x2 - i / 2 - 2, y1 + i);

}

**if** (remainTimes < 55) {

**for** (**int** i = 0; i < remainTimes; i++) {

g.drawLine(x1 + i / 2 + 2, y2 - i - 1, x2 - i / 2 - 2, y2 - i

- 1);

}

g.drawLine((x1 + x2) / 2, (y1 + y2) / 2, (x1 + x2) / 2, y2 - 2);

g.drawLine((x1 + x2) / 2 + 1, (y1 + y2) / 2 + 1, (x1 + x2) / 2 + 1,y2 - 2);//两条竖线

g.setColor(getBackground());

**for** (**int** i = 0; i < remainTimes; i++) {

g.drawLine(x1 + i / 2 + 2, y1 + i, x2 - i / 2 - 2, y1 + i);//覆盖上边的倒三角

}

}

**if** (remainTimes >= 50 && remainTimes <= 55)

overJLabel.setText(55-remainTimes +"s");

**if** (remainTimes == 56)

overJLabel.setText("OVER");

}

**public** **void** setTimes(**int** x) {

**this**.remainTimes = x;

}

**public** **int** getTimes() {

**return** remainTimes;

}

**public** **void** run() {

**while** (dijiguan < 20) {

**if** (remainTimes == 0) {

JOptionPane.*showMessageDialog*(**null**, "ready？");

}

**if** (remainTimes == 56) {

JOptionPane.*showMessageDialog*(**null**, "time out！game over！");

}

remainTimes++;

repaint();

**try** {

**if** (dijiguan < 6)

Thread.*sleep*(1500 - dijiguan \* 100);

**if** (dijiguan >= 6 && dijiguan <= 8)

Thread.*sleep*(1000 - (dijiguan - 5) \* 50);

**if** (dijiguan > 8)

Thread.*sleep*(850 - (dijiguan - 8) \* 20);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

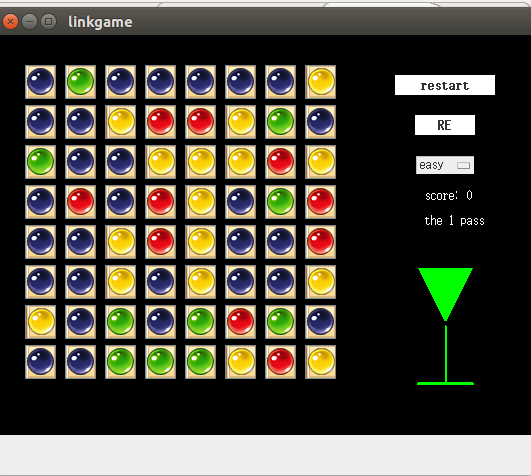
}

## 游戏结束

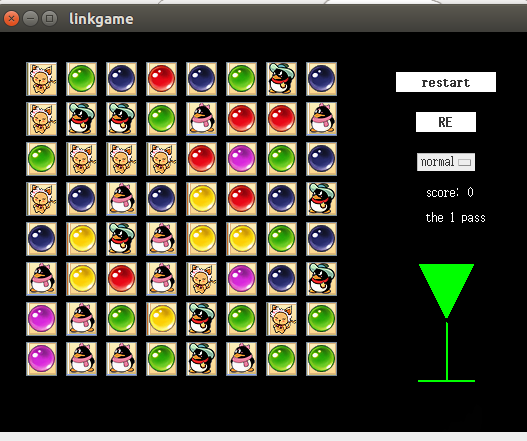


## 不同难易程度的游戏界面

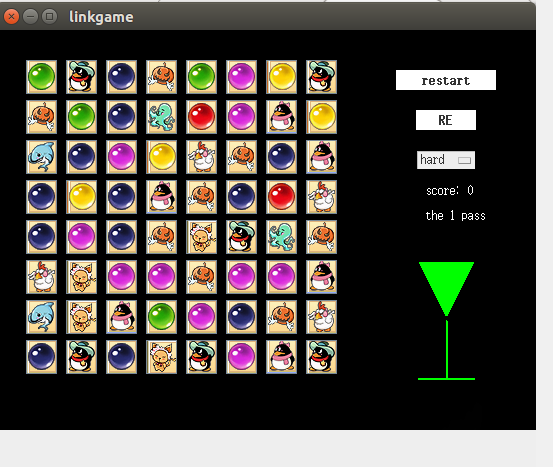
### 简单



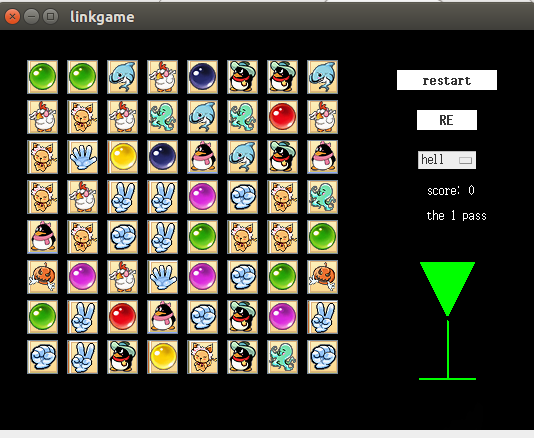
### 一般



### 困难



### 变态难



# 实验感想

本次大作业编写了经典小游戏连连看，通过不断地改正去达到相应的功能，我在这个过程中也学习到了很多，不足之处在于游戏界面不太美观，计时沙漏过于简单，总体来说还不错，感谢老师本学期教授了许多Linux操作系统的知识，让我在本学期受益匪浅。