2048游戏设计与实现

课题名称：2048游戏开发与测试

院系：电气学院

专业：电气工程及其自动化

班级：电气48

小组成员：赵红阳 朱正昊 陶站平

1. **摘要**

在生活节奏不断加快的现代社会，轻松的益智游戏对缓解生活压力，调节情绪具有重要的意义。《2048》是比较流行的一款[数字游戏](http://baike.baidu.com/view/348101.htm" \t "_blank)。原版2048首先在GitHub上发布，原作者是Gabriele Cirulli，是基于《1024》和《[小3传奇](http://baike.baidu.com/view/12239700.htm" \t "_blank)》的玩法开发而成的新型数字游戏。而游戏设计初衷是一款益智类的游戏，其特点在于轻松，简单，因此，开发要求做到各项功能要完备、操作要简便、易学易用。

本文采用c#语言开发了2048小游戏，通过windows控制台应用程序实现了2048的基本运行，而后进过图形界面的处理，以不同的数字西安市不同的颜色，并拥有了成绩记录，成绩截图的各项功能。本着简单，易于操作的原则和缓解生活压力、降低生活节奏的初衷，本小组在游戏界面中添加了帮助菜单和各类提示，让游戏的用户界面更加人性化。

1. **关键词**

2048 c# 休闲小游戏

**[一、 游戏开发环境 4](#_Toc31202)**

[1． 开发工具 4](#_Toc7165)

[2． 应用环境 6](#_Toc11784)

**[二、 游戏需求分析 6](#_Toc31110)**

[1. 需求分析 6](#_Toc4638)

[2. 可行性分析 6](#_Toc5967)

**[三、 程序概要设计 7](#_Toc6719)**

**[四、 程序详细设计 8](#_Toc7058)**

[1. 游戏主界面 8](#_Toc29732)

[2. 游戏结束界面 18](#_Toc9215)

[3. 新纪录界面 19](#_Toc14887)

[4. 主要功能代码 22](#_Toc20272)

**[五、 游戏测试 32](#_Toc28548)**

[1． 测试的意义 32](#_Toc32366)

[2． 测试过程 32](#_Toc5996)

[3． 测试内容 33](#_Toc31470)

[4． 测试结果 33](#_Toc1511)

1. 游戏开发环境
2. 开发工具

C#(读做 "C sharp")暂时没有中文译名，专业人士一般读"C sharp"，现在很多非专业一般读"C井"。

　　C#是一种安全的、稳定的、简单的、优雅的，由C和C++衍生出来的面向对象的编程语言。它在继承C和C++强大功能的同时去掉了一些它们的复杂特性（例如没有宏和模版，不允许多重继承）。C#综合了VB简单的可视化操作和C++的高运行效率，以其强大的操作能力、优雅的语法风格、创新的语言特性和便捷的面向组件编程的支持成为.NET开发的首选语言。

并且C#成为ECMA与ISO标准规范。C#看似基于C++写成，但又融入其它语言如Pascal、Java、VB等。

Microsoft在正式的场合把C#描述为一种简单、现代、面向对象、类型非常安全、派生于C和C++的编程语言。大多数独立的评论员对其说法是“派生于C、C++和Java”。这种描述在技术上是非常准确的，但没有涉及到该语言的真正优点。从语法上看，C#非常类似于C和Java，许多关键字都是相同的，C#也使用类似于C和C++的块结构，并用括号({})来标识代码块，用分号分隔各行语句。对C#代码的第一印象是它非常类似于C和Java代码。但在这些表面上的类似性后面，C#学习起来要比C++容易得多，但比Java难一些。其设计与现代开发工具的适应性要比其他语言更高，它同时具有Visual Basic的易用性、高性能以及C++的低级内存访问性。

Ｃ#包括以下一些特性：

* 完全支持类和面向对象编程，包括接口和继承、虚函数和运算符重载的处理。
* 定义完整、一致的基本类型集。
* 对自动生成XML文档说明的内存。
* 自动清理动态分配的内存。
* 可以用用户定义的特性来标记类或方法
* 对.NET基类库的完全访问权，并易于访问Windows API。
* 可以使用指针和直接内存访问，但C#语言可以在没有它们的条件下访问内存。
* 以VB的风格支持属性和事件。
* 改变编译器选项，可以把程序编译为可执行文件或.NET组件库，该组件库可以用与ActiveX控件(COM组件)相同的方式由其他代码调用。
* C#可以用于编写ASP.NET动态Web页面和XML Web服务。

1. 应用环境

操作系统：Windows 7/8/8.1

应用软件：Microsoft Visual Studio 2012/2013

1. 游戏需求分析
2. 需求分析

现代社会对休闲小游戏的需求是：提高游戏的操作可行性，降低游戏的操作难度，降低游戏的上手难度，降低游戏的竞争需求，对使用设备的要求趋于简单化和多样化，游戏界面更加人性化，功能更加贴合使用。

1. 可行性分析
2. 技术可行性

根据游戏设计的要求，可以通过c#进行源代码的编辑，通过windows界面进行用户界面的编辑和优化，结合时间和目前学习水平等各项因素，项目所要求功能和性能完全可以实现。

1. 运行可行性

游戏基本要求是.net framework4.0及以上，在大部分的用户设备上可以实现，并且游戏运行对硬件几乎无要求，项目运行可以得到保证。

1. 可行性综合分析

综合以上两点，项目开发可以得到技术和运行的保证，证明项目可行。

1. 程序概要设计

退出

重启

游戏结束

退出游戏

打开帮助

各项功能实现

Esc

各功能键

F1

键盘控制方块移动

游戏运行

启动游戏界面

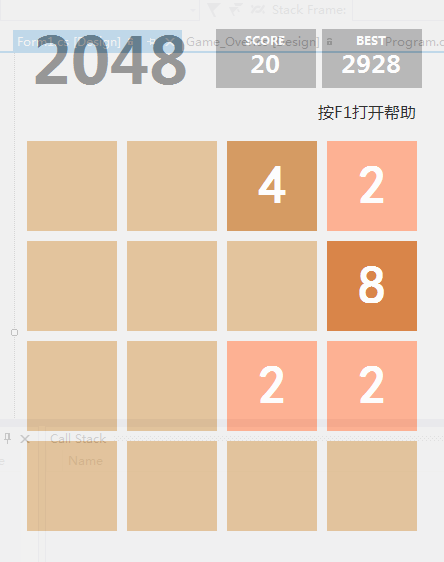
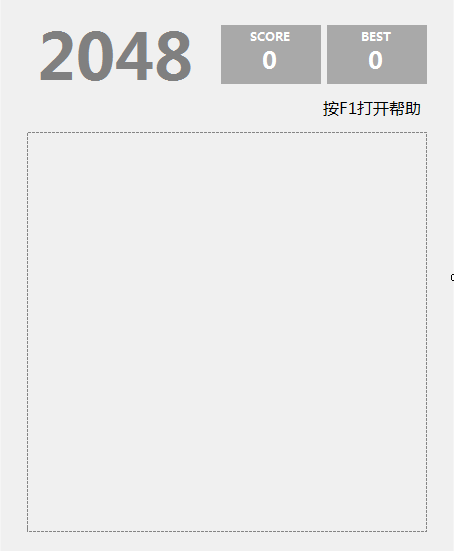
打开帮助对话框

检测是否为第一次启动

游戏启动

1. 程序详细设计
2. 游戏主界面

截图如下：



其功能代码如下：

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Runtime.Serialization;

using System.Runtime.Serialization.Formatters.Binary;

using System.IO;

namespace \_2048

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

Class1 c;

\_2048messageBox mes = new \_2048messageBox();

Bitmap bit = new Bitmap(400, 400);

private void Form1\_Load(object sender, EventArgs e)

{

if (File.Exists("D:\\2048记录")) //监测是否有存档

{

ToLoad();

}

else //如果没有存档，生成新的游戏。

{

\_2048messageBox mes1 = new \_2048messageBox();

mes1.a = "提示";

mes1.b = " 检测到您是第一次打开，可按F1打开帮助。\r\n 游戏为无限模式，直到GameOver才会终止。";

mes1.StartPosition = FormStartPosition.CenterScreen;

mes1.ShowDialog();

c = new Class1();

c.Reset();

}

mes.a = "提示"; //生成帮助菜单

mes.b = "F2:窗口总在最前\r\nF3:解锁/锁定窗口\r\nF4:隐藏任务栏图标\r\nF5:重新开始\r\nF6:截图并保存\r\nShife + ↑↓：调整透明度\r\n↑↓←→：控制方块移动\r\nESC:退出";

drow();

Num\_pictureBox.Refresh();

} //窗口加载前的操作

private void Form1\_KeyDown(object sender, KeyEventArgs e)

{

switch (e.KeyCode)

{

case Keys.Up:

if (e.Modifiers == Keys.Shift)

this.Opacity -= 0.1;

else

{

c.Up();

if (c.change)

c.Add();

}

break;

case Keys.Down:

if (e.Modifiers == Keys.Shift)

this.Opacity += 0.1;

else

{

c.Down();

if (c.change)

c.Add();

}

break;

case Keys.Left:

c.Left();

if (c.change)

c.Add();

break;

case Keys.Right:

c.Right();

if (c.change)

c.Add();

break;

case Keys.F1:

mes.ShowDialog();

break;

case Keys.F2:

this.TopMost = !this.TopMost;

break;

case Keys.F3:

if (this.FormBorderStyle == FormBorderStyle.FixedToolWindow)

this.FormBorderStyle = FormBorderStyle.None;

else

this.FormBorderStyle = FormBorderStyle.FixedToolWindow;

break;

case Keys.F4:

this.ShowInTaskbar = !this.ShowInTaskbar;

break;

case Keys.F5:

c = new Class1();

c.Reset();

Num\_pictureBox.Refresh();

break;

case Keys.F6:

\_2048Screen();

\_2048messageBox mes2 = new \_2048messageBox();

mes2.a = "保存成功";

mes2.b = "保存在" + Directory.GetCurrentDirectory() + "\\成绩截图.bmp";

mes2.ShowDialog();

break;

case Keys.Escape:

this.Close();

break;

}

drow();

Num\_pictureBox.Refresh();

grade.Text = c.grade.ToString();

if (c.die)

{

gameOver();

}

} //捕捉键盘按下的键，做出相应操作

private void display(int m, Point dian)

{

Graphics gra = Graphics.FromImage(bit);

switch (m)

{

case 0:

{ gra.FillRectangle(new SolidBrush(Color.BurlyWood), dian.X, dian.Y, 90, 90); } break;

case 2:

{ gra.FillRectangle(new SolidBrush(Color.LightSalmon), dian.X, dian.Y, 90, 90); } break;

case 4:

{ gra.FillRectangle(new SolidBrush(Color.Peru), dian.X, dian.Y, 90, 90); } break;

case 8:

{ gra.FillRectangle(new SolidBrush(Color.Chocolate), dian.X, dian.Y, 90, 90); } break;

case 16:

{ gra.FillRectangle(new SolidBrush(Color.Gray), dian.X, dian.Y, 90, 90); } break;

case 32:

{ gra.FillRectangle(new SolidBrush(Color.DarkSeaGreen), dian.X, dian.Y, 90, 90); } break;

case 64:

{ gra.FillRectangle(new SolidBrush(Color.Gold), dian.X, dian.Y, 90, 90); } break;

case 128:

{ gra.FillRectangle(new SolidBrush(Color.HotPink), dian.X, dian.Y, 90, 90); } break;

case 256:

{ gra.FillRectangle(new SolidBrush(Color.DarkOrange), dian.X, dian.Y, 90, 90); } break;

case 512:

{ gra.FillRectangle(new SolidBrush(Color.LightPink), dian.X, dian.Y, 90, 90); } break;

case 1024:

{ gra.FillRectangle(new SolidBrush(Color.DarkRed), dian.X, dian.Y, 90, 90); } break;

case 2048:

{ gra.FillRectangle(new SolidBrush(Color.Red), dian.X, dian.Y, 90, 90); } break;

}

switch (m)

{

case 2:

case 4:

case 8:

gra.DrawString(m.ToString(), new Font("黑体", 40.5f, FontStyle.Bold), new SolidBrush(Color.White), dian.X + 22, dian.Y + 17);

break;

case 16:

case 32:

case 64:

gra.DrawString(m.ToString(), new Font("黑体", 40.5f, FontStyle.Bold), new SolidBrush(Color.White), dian.X + 8, dian.Y + 17);

break;

case 128:

case 256:

case 512:

gra.DrawString(m.ToString(), new Font("黑体", 35.5f, FontStyle.Bold), new SolidBrush(Color.White), dian.X + 0, dian.Y + 20);

break;

case 1024:

case 2048:

case 4096:

case 8192:

gra.DrawString(m.ToString(), new Font("黑体", 30.5f, FontStyle.Bold), new SolidBrush(Color.White), dian.X - 4, dian.Y + 23);

break;

}

} //画出一个方块和方块上的数字

private void drow()

{

for (int x = 1; x <= 4; x++)

{

for (int y = 1; y <= 4; y++)

{

Point p = new Point(x \* 100 - 95, y \* 100 - 95);

display(c.i[x, y], p);

}

}

} //画出每个方块

private void gameOver()

{

if (c.bestGrade < c.grade) //判断本次成绩是否刷新纪录

{

c.bestGrade = c.grade;

bestgrade.Text = c.bestGrade.ToString();

\_2048Screen();

\_2048messageBox mes3 = new \_2048messageBox();

mes3.a = "恭喜！";

mes3.b = "新的记录！自动为您保存截图。\r\n保存在" + bitfile;

mes3.ShowDialog();

c.Reset();

drow();

Num\_pictureBox.Refresh();

}

else

{

Game\_Over g = new Game\_Over();

g.bg = c.bestGrade;

g.g = c.grade;

DialogResult d = g.ShowDialog();

switch (d)

{

case DialogResult.Retry:

c.Reset();

drow();

Num\_pictureBox.Refresh();

grade.Text = c.grade.ToString();

bestgrade.Text = c.bestGrade.ToString();

break;

case DialogResult.Abort:

\_2048Screen();

\_2048messageBox mes2 = new \_2048messageBox();

mes2.a = "保存成功";

mes2.b = "保存在" + bitfile;

mes2.ShowDialog();

c.Reset();

classSave();

drow();

Num\_pictureBox.Refresh();

break;

case DialogResult.No:

c.Reset();

this.Close();

break;

}

}

} //游戏结束时需要的操作

private string bitfile = Directory.GetCurrentDirectory() + "\\成绩截图";//成绩截图的目录

private void \_2048Screen()

{

Bitmap b = new Bitmap(this.Width, this.Height);

Graphics gr = Graphics.FromImage(b);

gr.CopyFromScreen(this.Location, new Point(0, 0), this.Size);

gr.Dispose();

if (!File.Exists(bitfile)) //判断截图是否已存在，如果存在，在路径后面加上一个空格继续保存，避免覆盖

bitfile += " ";

b.Save(bitfile+".bmp");

} //截图

private void Form1\_FormClosed(object sender, FormClosedEventArgs e)

{

classSave();

}//窗口关闭后需要的操作

private void Num\_pictureBox\_Paint(object sender, PaintEventArgs e)

{

Num\_pictureBox.BackgroundImage = bit;

grade.Text = c.grade.ToString();

} //显示方块的主界面的重绘事件，其中加入了刷新当前成绩的语句，确保当前成绩及时刷新

private void classSave()

{

FileStream fw = new FileStream("D:\\2048记录", FileMode.Create, FileAccess.Write);

BinaryFormatter formatter\_w = new BinaryFormatter();

formatter\_w.Serialize(fw, c);

//File.SetAttributes("D:\\2048记录", FileAttributes.Hidden);

fw.Close();

}//序列化Class1类，并保存，相当于存档

private void ToLoad()

{

FileStream fr = new FileStream("D:\\2048记录", FileMode.Open, FileAccess.Read);

BinaryFormatter formatter\_r = new BinaryFormatter();

c = (Class1)formatter\_r.Deserialize(fr);

grade.Text = c.grade.ToString();

bestgrade.Text = c.bestGrade.ToString();

fr.Close();

}

private void Num\_pictureBox\_Click(object sender, EventArgs e)

{

}//从文件反序列化读取存档

//退出前加一个确认的选项

//private void Form1\_FormClosing(object sender, FormClosingEventArgs e)

//{

// DialogResult res = MessageBox.Show("是否退出？", "提示",

// MessageBoxButtons.OKCancel,MessageBoxIcon.Question);

// if (res == DialogResult.OK)

// {

// e.Cancel = false;

// }

// else

// {

// e.Cancel = true;

// }

//}

}

}

1. 游戏结束界面

截图如下：



其功能代码如下：

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace \_2048

{

public partial class Game\_Over : Form

{

public Game\_Over()

{

InitializeComponent();

}

public int g { get; set;}

public int bg { get; set; }

private void Game\_Over\_Load(object sender, EventArgs e)

{

label2.Text += g;

label3.Text += bg;

this.TopLevel = true;

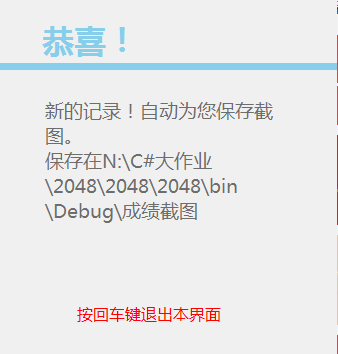
}

}

}

1. 新纪录界面

截图如下：



其功能代码如下：

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace \_2048

{

public partial class \_2048messageBox : Form

{

public \_2048messageBox()

{

InitializeComponent();

}

public string a { get; set; }

public string b { get; set; }

private void \_2048messageBox\_Load(object sender, EventArgs e)

{

label1.Text = a;

textBox1.Text = b;

this.TopLevel = true;

}

public void b\_add(string x)

{

textBox1.AppendText(x);

}

private void \_2048messageBox\_MouseClick(object sender, MouseEventArgs e)

{

this.Close();

}

private void \_2048messageBox\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

this.Close();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

}

}

1. 主要功能代码

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace \_2048

{

[Serializable]

class Class1

{

public int[,] i = new int[6, 6]; //使用6\*6数组的原因是可以简化避免监测是否Game Over时的数组索引报错

public int grade { get; private set; } //存储当前成绩

public int bestGrade { get ;set;} //存储最好成绩

private int quantity { get; set; } //当前有多少个方块

private Random ra = new Random();

public bool die = false; //是否Game Over

public bool change = false; //按下某个方向键后会记录方块可否移动

public void Reset() //刷新所有方块 重新开始

{

for (int x = 0; x <= 5; x++)

for (int y = 0; y <= 5; y++)

i[x, y] = 0; //方块归零

quantity = 0; //方块计数归零

die = false; //重置游戏状态

if (grade > bestGrade)

bestGrade = grade; //记录最好成绩

grade = 0; //成绩归零

Add();

Add(); //为新游戏增加两个方块

}

public void Add() //增加新方块

{

int x = ra.Next(1, 5);

int y = ra.Next(1, 5); //随机选定位置

if (i[x, y] == 0)

{

if (ra.Next(1, 101) >= 90)

i[x, y] = 4; //有10%的几率刷新出方块4

else i[x, y] = 2; //有90%的几率刷新出方块2

quantity++; //增加一个新方块后计数器加1

Die(); //判断游戏是否已经死局

}

else Add(); //如果随机选定的位置不能添加新的放开，则重新调用该方法

}

#region 上下左右——非零元素处理 上下左右处理时采用了先处理0元素，再处理非0元素的方法

public void Down()

{

change = false;

down(); //处理零方块，其他方法同

for (int x = 1; x <= 4; x++)

{

if (i[x, 4] == i[x, 3] && i[x, 4] + i[x, 3] != 0)

{

if (i[x, 2] == i[x, 1])

{

i[x, 4] \*= 2;

i[x, 3] = i[x, 2] \* 2;

i[x, 2] = 0;

i[x, 1] = 0;

grade += i[x, 4] + i[x, 3];

}

else

{

i[x, 4] \*= 2;

i[x, 3] = i[x, 2];

i[x, 2] = i[x, 1];

i[x, 1] = 0;

grade += i[x, 4];

}

change = true;

}

else if (i[x, 3] == i[x, 2] && i[x, 3] + i[x, 2] != 0)

{

i[x, 3] \*= 2;

i[x, 2] = i[x, 1];

i[x, 1] = 0;

change = true;

grade += i[x, 3];

}

else if (i[x, 2] == i[x, 1] && i[x, 2] + i[x, 1] != 0)

{

i[x, 2] \*= 2;

i[x, 1] = 0;

change = true;

grade += i[x, 2];

}

}

GetQuantity();

}

public void Up()

{

change = false;

up();

for (int x = 1; x <= 4; x++)

{

if (i[x, 1] == i[x, 2] && i[x, 1] + i[x, 2] != 0)

{

if (i[x, 3] == i[x, 4]）

{

i[x, 1] \*= 2;

i[x, 2] = i[x, 3] \* 2;

i[x, 3] = 0;

i[x, 4] = 0;

grade += i[x, 1] + i[x, 2];

}

else

{

i[x, 1] \*= 2;

i[x, 2] = i[x, 3];

i[x, 3] = i[x, 4];

i[x, 4] = 0;

grade += i[x, 1];

}

change = true;

}

else if (i[x, 2] == i[x, 3] && i[x, 2] + i[x, 3] != 0)

{

i[x, 2] \*= 2;

i[x, 3] = i[x, 4];

i[x, 4] = 0;

change = true;

grade += i[x, 2];

}

else if (i[x, 3] == i[x, 4] && i[x, 3] + i[x, 4] != 0)

{

i[x, 3] \*= 2;

i[x, 4] = 0;

change = true;

grade += i[x, 3];

}

}

GetQuantity();

}

public void Left()

{

change = false;

left();

for (int y = 1; y <= 4; y++)

{

if (i[1, y] == i[2, y] && i[1, y] + i[2, y] != 0)

{

if (i[3, y] == i[4, y])

{

i[1, y] \*= 2;

i[2, y] = i[3, y];

i[3, y] = 0;

i[4, y] = 0;

grade += i[1, y] + i[2, y];

}

else

{

i[1, y] \*= 2;

i[2, y] = i[3, y];

i[3, y] = i[4, y];

i[4, y] = 0;

grade += i[1, y];

}

change = true;

}

else if (i[2, y] == i[3, y] && i[2, y] + i[3, y] != 0)

{

i[2, y] \*= 2;

i[3, y] = i[4, y];

i[4, y] = 0;

change = true;

grade += i[2, y];

}

else if (i[3, y] == i[4, y] && i[3, y] + i[4, y] != 0)

{

i[3, y] \*= 2;

i[4, y] = 0;

change = true;

grade += i[3, y];

}

}

GetQuantity();

}

public void Right()

{

change = false;

right();

for (int y = 1; y <= 4; y++)

{

if (i[4, y] == i[3, y] && i[4, y] + i[3, y] != 0)

{

if (i[2, y] == i[1, y])

{

i[4, y] \*= 2;

i[3, y] = i[2, y];

i[2, y] = 0;

i[1, y] = 0;

grade += i[4, y] + i[3, y];

}

else

{

i[4, y] \*= 2;

i[3, y] = i[2, y];

i[2, y] = i[1, y];

i[1, y] = 0;

grade += i[4, y];

}

change = true;

}

else if (i[3, y] == i[2, y] && i[3, y] + i[2, y] != 0)

{

i[3, y] \*= 2;

i[2, y] = i[1, y];

i[1, y] = 0;

change = true;

grade += i[3, y];

}

else if (i[2, y] == i[1, y] && i[3, y] + i[1, y] != 0)

{

i[2, y] \*= 2;

i[1, y] = 0;

change = true;

grade += i[2, y];

}

}

GetQuantity();

}

private void down() //先将方块移到同一边,其他方法同

{

for (int x = 1; x <= 4; x++)

{

if (i[x, 4] == 0 && i[x, 1] + i[x, 2] + i[x, 3] != 0)

{

i[x, 4] = i[x, 3];

i[x, 3] = i[x, 2];

i[x, 2] = i[x, 1];

i[x, 1] = 0;

change = true;

down();

}

else if (i[x, 3] == 0 && i[x, 1] + i[x, 2] != 0)

{

i[x, 3] = i[x, 2];

i[x, 2] = i[x, 1];

i[x, 1] = 0;

change = true;

down();

}

else if (i[x, 2] == 0 && i[x, 1] != 0)

{

i[x, 2] = i[x, 1];

i[x, 1] = 0;

change = true;

}

}

}

private void up()

{

for (int x = 1; x <= 4; x++)

{

if (i[x, 1] == 0 && i[x, 4] + i[x, 3] + i[x, 2] != 0)

{

i[x, 1] = i[x, 2];

i[x, 2] = i[x, 3];

i[x, 3] = i[x, 4];

i[x, 4] = 0;

change = true;

up();

}

else if (i[x, 2] == 0 && i[x, 4] + i[x, 3] != 0)

{

i[x, 2] = i[x, 3];

i[x, 3] = i[x, 4];

i[x, 4] = 0;

change = true;

up();

}

else if (i[x, 3] == 0 && i[x, 4] != 0)

{

i[x, 3] = i[x, 4];

i[x, 4] = 0;

change = true;

}

}

}

private void left()

{

for (int y = 1; y <= 4; y++)

{

if (i[1, y] == 0 && i[4, y] + i[3, y] + i[2, y] != 0)

{

i[1, y] = i[2, y];

i[2, y] = i[3, y];

i[3, y] = i[4, y];

i[4, y] = 0;

change = true;

left();

}

else if (i[2, y] == 0 && i[4, y] + i[3, y] != 0)

{

i[2, y] = i[3, y];

i[3, y] = i[4, y];

i[4, y] = 0;

change = true;

left();

}

else if (i[3, y] == 0 && i[4, y] != 0)

{

i[3, y] = i[4, y];

i[4, y] = 0;

change = true;

}

}

}

private void right()

{

for (int y = 1; y <= 4; y++)

{

if (i[4, y] == 0 && i[1, y] + i[2, y] + i[3, y] != 0)

{

i[4, y] = i[3, y];

i[3, y] = i[2, y];

i[2, y] = i[1, y];

i[1, y] = 0;

change = true;

right();

}

else if (i[3, y] == 0 && i[1, y] + i[2, y] != 0)

{

i[3, y] = i[2, y];

i[2, y] = i[1, y];

i[1, y] = 0;

change = true;

right();

}

else if (i[2, y] == 0 && i[1, y] != 0)

{

i[2, y] = i[1, y];

i[1, y] = 0;

change = true;

}

}

}

#endregion

private void GetQuantity() //获得当前游戏中的方块数目

{

int count = 0; //初始化计数器

for (int x = 1; x <= 4; x++)

for (int y = 1; y <= 4; y++)

{

if (i[x, y] != 0)

count++; //逐块检索，每检索到一个方块计数器加1

}

quantity = count; //记录当前方块数目

}

private void Die() //判断当前是否死局

{

int count = 0; //初始化计数器

if (quantity == 16) //若游戏中方块数达到16个，则开始判断是否死局

{

for (int x = 1; x <= 3; x += 2)

for (int y = 1; y <= 3; y += 2)

if (!GetEqual(x, y))

count++;

for (int x = 2; x <= 4; x += 2)

for (int y = 2; y <= 4; y += 2)

if (!GetEqual(x, y))

count++; //选取了所有方块中互不相邻的八块，判断是否可加

if (count == 8)

die = true; //若计数器达到8，则说明所有临近方块均不可加，游戏死局

}

}

private bool GetEqual(int x, int y) //判断方块是否与周围方块相等（是否可加）

{

if (i[x, y] == i[x - 1, y])

return true;

else if (i[x, y] == i[x + 1, y])

return true;

else if (i[x, y] == i[x, y - 1])

return true;

else if (i[x, y] == i[x, y + 1])

return true; //若与上下左右其中一个相等，则可加

else return false; //否则不可加

}

}

}

1. 游戏测试
2. 测试的意义

游戏测试是为了发现错误而执行程序的过程，成功的测试是发现了至今尚未发现的错误的测试。测试的目的就是希望能以最少的人力和时间发现潜在的各种错误和缺陷。游戏测试是保证系统质量和可靠性的关键步骤，是对游戏开发过程中的系统分析游戏设计和实施的最后复查。

1. 测试过程
2. 拟定测试计划。根据游戏的总体情况，仔细分析游戏各个步骤的运行可能出现的状况，对各种情况进行汇总，然后拟定出一份具体测试步骤。
3. 初步测试。根据拟定好的计划进行游戏调试，出现意外情况时及时记录。测试完后对所记录的意外情况进行分析，然后提出改进的方案，再进行修改。
4. 深度测试。对游戏进行最后的修改确认没有漏洞后再进行测试，从各个方面进行一次整体的排查，知道不再出现意料之外的情况。
5. 测试人员。测试初级阶段主要由游戏主要编程员进行测试，以便能够全面的进行一次错误排查。然后由不同的玩家进行测试，集众人之益，能够从不同的角度发现问题，以便更好地改进。
6. 测试内容

（1）初始界面

打开游戏后，会弹出初始界面，显示按回车键退出此界面

（2）帮助界面

按回车后，进入正式游戏界面，可以使用F1键打开帮助，这时会弹出小对话框，提示各按键相应功能。再次按回车键可以退出。

（3）正式游戏界面

每次可以选择上下左右其中一个方向去滑动，每滑动一次，所有的数字方块都会往滑动的方向靠拢外，系统也会在空白的地方乱数出现一个数字方块，相同数字的方块在靠拢、相撞时会相加。不断的叠加最终拼凑出2048这个数字就算成功。但此时游戏不会停止，会继续直到无路可走，当游戏结束时，会自动截图，并将成绩保存在文件夹内。

1. 测试结果

程序运行正常，没有发现什么错误。