

**课 程 设 计 报 告**

**题目：《2048小游戏》的设计和实现**

**课程名称： 程序设计基础综合课程设计**

**专业班级： 信息安全1602班**

**学 号： U201614848**

**姓 名： 郭倜维**

**指导教师： 卢萍**

**报告日期：**

**计算机科学与技术学院**

**课程设计任务书**

题 目 《2048小游戏》的设计和实现

|  |
| --- |
| 课题内容：  《2048》是 2014 年发布了一款风靡全球的数字小游戏（你玩过吗？它的发  明者是意大利的 19 岁男生戚鲁利）。游戏规则很简单：  （1）开始时棋盘内随机出现两个数字，出现的数字仅可能为 2 或 4，如图 1 所示：    图 1 一个起始棋局  （2）玩家可以选择上、下、左、右四个方向移动。每选择一次移动，棋盘上的所有数字都沿同一方向（在同行或同列上沿该方向）移动。若棋盘内的一个或多个数字出现位移或合并，视为有效移动。  位移：在选择的移动方向（不管是行还是列）上，若前方有空格，则数字沿该方向向前移动，称为位移。如果没有空格，但有相邻的相同数字，则先进行数字合并，然后后方的数字向前移位。  数字合并：若在选择的移动方向上有相同的数字，则进行合并（即两个相同的数字相加，用其和取代原来的两个数字）。每次有效移动可以同时合并，即如果同一行或列（或多行或列）中有多对相同的数字，则两两同时相加；但不可以连续合并，即若在同一行或列的移动方向上有 3 个或 4 个相同的数字，不能连续相加，如 2←2←2，则合并为 4|2，而不是 3 个 2 相加；同样，也不能把合并的和与后面的数字连续相加，如 2←2←4，合并后为 4|4，而不是 8）。    得分：每次合并所得的所有新生成数字的和即为该步的有效得分，如当前移动只有一次 2+2 的合并，则该次移动得 4 分，而如果有两个 2+2 的合并则得 8 分。  填补新数字：每有效移动一步（即有移位或合并发生的情况下），会在棋盘  的空位（指无数字的位置）随机出现一个数字（这个数字只能是 2 或 4）。新出现的数字和棋盘上原有的数字一起参与后面的移动。如图 2 所示。    图 2 某个时刻棋盘的状态  （4）若游戏的过程中，使得棋盘上的某个位置出现了数字 2048，则判胜， 游戏结束（如图 3（a））。如果棋盘被数字填满，无法进行有效移动了，则判负，游戏结束（如图 3（b））。    （a）获胜 （b）失败  图 3 游戏结果  本题目的任务是理解上述游戏规则，自行设计和开发一款《2048 小游戏》的程序。 |
| 课题任务要求  本题目的设计分为两部分：基础部分和提高部分。  1．基础部分  基于 PC 平台，编程实现《2048 小游戏》基本程序。要求：   1. 完成软件程序的开发，玩家可以运行程序进行游戏。 2. 具有图形用户界面，以图形的方式展示棋盘布局和数字移位、合并的过程；界面简洁、美观，操作流畅，具有良好的用户体验。 3. 为玩家建立账号，实现计分功能，为玩家记录每次游戏的成绩，并能   够对每位玩家的最好历史成绩进行排名。   1. 能够正确判断获胜或失败棋局，并结束游戏。   2．提高部分  在基础部分的基础上，增加以下功能：   1. 实现悔棋（倒退）和重复（前进）功能。不限制悔棋步数，直至起始棋局。 2. 记录每次游戏操作的轨迹，实现历史回放功能。   设计走棋算法，实现计算机控制下的自动下棋功能。能够对当前棋局进行判别，按照高得分并取胜的目标规划走棋步骤；能够进行一定的机器学习，提高计算机自动下棋水平。 |
| 主要参考文献：  [1]曹计昌,卢萍,李开. C语言与程序设计. 北京：电子工业出版社,2013.1  [2]李开,卢萍,曹计昌. C语言实验与课程设计.北京：科学出版社,2011.3  [3]霍亚飞. Qt Creator 快速入门. 北京：北京航空航天大学出版社，2017.1  [4]金大. Qt 5 开发实践. 北京：人民邮电出版社，2015.9  [5]严蔚敏，吴伟民. 数据结构. 北京：清华大学出版社 2004.2 |

**目录**

[1 绪言 6](#_Toc509194064)

[1.1 课题背景 6](#_Toc509194065)

[1.2 课题研究的目的及意义 6](#_Toc509194066)

[1.3 国内外概况 6](#_Toc509194067)

[2 系统设计方案的研究 6](#_Toc509194068)

[2.1 系统的控制特点与性能要求 6](#_Toc509194069)

[2.2 系统实现的原理 6](#_Toc509194070)

[2.3 系统实现方案分析 6](#_Toc509194071)

[2.4 方案的分析比较 6](#_Toc509194072)

[3 系统各模块的设计 6](#_Toc509194073)

[3.1 系统各模块设计方式的选用 6](#_Toc509194074)

[3.2 系统各模块的功能设计 6](#_Toc509194075)

[4 系统各模块的功能实现 6](#_Toc509194076)

[4.1 系统各模块的功能阐述 6](#_Toc509194077)

[4.2 系统各模块的功能实现 6](#_Toc509194078)

[5 总结与展望 7](#_Toc509194079)

[参考文献 7](#_Toc509194080)

[附录 7](#_Toc509194081)

# 1 绪言

## 课题背景

《2048》是一款现象级的游戏，其在全球的风靡程度甚至超越了《Flappy Bird》，有超过2300万人玩过或者正在玩这款游戏。

## 1.2 课题研究的目的

学生通过课堂，学习并掌握了一定的程序编写知识，包括高级程序语言的重要语法，例如C语言等，基本算法以及基本数据结构的运用。

通过对该课题的开发，提升学生的软件综合设计能力。该课题研究目的主要在三个方面。

其一，在程序设计方面，强化学生高级程序语言的运用能力，进一步掌握动态内存管理、文件操作的技术，同时掌握图形用户界面的开发技术和复杂程序的构造与编制技术等。

其二，在数据结构方面，加深对线性表、树、图等基本数据结构的理解，强化这些结构的应用，并结合课题的需要，设计更为高级有效的数据结构， 在这个过程中提高理论联系实际的能力。

其三加强算法设计训练，熟练掌握查找、排序及基本数据结构的操作算法。

## 1.3 国内外概况

2014年Gabriele Cirulli利用周末的时间完成了2048这个游戏的程序，他写这个游戏程序是因为想用一种不同的视觉展现效果和更快速的动画来创造属于他自己的游戏版本。这款游戏发行后不久，它就开始风靡全球，成为了一款现象级的游戏，甚至超越了“flappy bird”。

随后，Gabriele Cirulli把源代码对外开放，于是，在人们对这个游戏高热情的基础上，越来越多的人展开对这个游戏的探索与研究。

如今，国内外通过对该游戏多方面的研究，在原程序的基础上开发出了更多不同版本的该游戏；其中有传统版2048、汉服版2048（如图1.1）、六边形版2048（如图1.2）、朝代版2048（如图1.4）以及3D版2048（如图1.3）等。

这些版本游戏都是在原游戏的基础上进行了改进，汉服版本和朝代版本对数字累加的形式进行了修改，将数字转变为汉服和朝代，使得游戏看起来更加有趣和美观；3D版本将游戏界面3D化，方块之间的碰撞和交互更加刺激直观，支持360并无死角的视角旋转、移动及缩放，玩法新颖独特。；六边形版本将以往的16格构成的正方形转变为由18圆形构成的六边形，使得移动过程变得更加多样化。

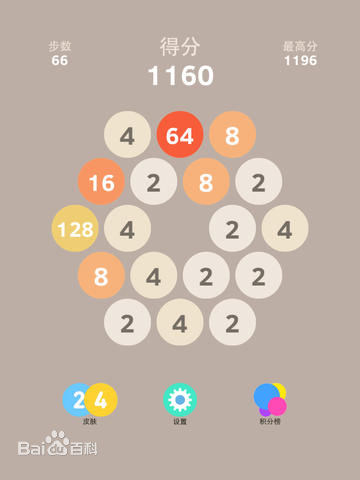
[](https://baike.baidu.com/pic/2048/13383511/0/a2cc7cd98d1001e97218060cb10e7bec54e79788?fr=lemma&ct=single) 

图1.1 汉服版2048 图1.2 六边形版2048

[](https://baike.baidu.com/pic/2048/13383511/0/79f0f736afc3793152a82d6be8c4b74542a911ab?fr=lemma&ct=single) [](https://baike.baidu.com/pic/2048/13383511/0/d31b0ef41bd5ad6ee853cfa882cb39dbb7fd3ccd?fr=lemma&ct=single)

图1.3 3D版2048 图1.4 朝代版2048

# 2 系统设计方案的研究

## 2.1 系统的控制特点

该版本的2048系统的控制特点体现在多个方面，

其一，该系统通过连接各个界面来控制各界面之间的跳转。在QT中，源程序通过对每个界面的特定按钮添加槽函数，然后在槽函数中完成关闭该界面打开新界面的功能来实现跳转界面，完成打开新窗口但不关闭该界面的功能来实现弹出提示窗口。如图2.1，该界面是游戏的首页，在首页中，有两个选择按钮，玩家可以通过选择“登录”，界面就会跳转到如图2.2的登录界面，这是通过控制按钮来实现对背后的槽函数控制，从而实现对页面跳转的控制。如图2.3，该界面是游戏的主界面也就是游戏界面，在该界面的左上角存在菜单栏，如图2.4；玩家可以选择特定菜单项获得自己需要的信息，比如选择“Rules”，就会在主界面之上弹出如图2.5的规则提示窗口；这就是通过控制按钮来完成对背后的槽函数的控制，从而实现窗口弹出的控制。

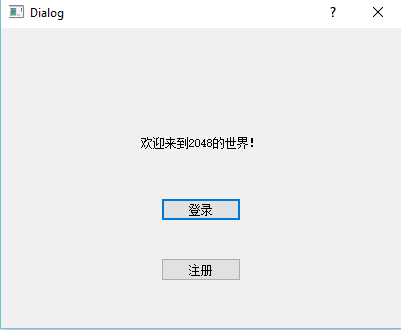


图2.1 首页



图2.2 登录界面

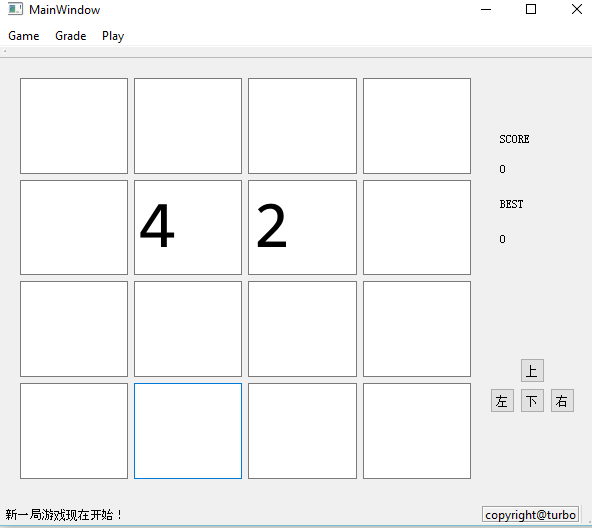


图2.3 游戏界面

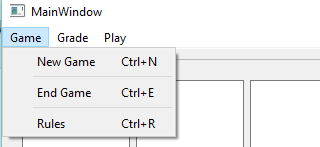


图2.4 菜单栏

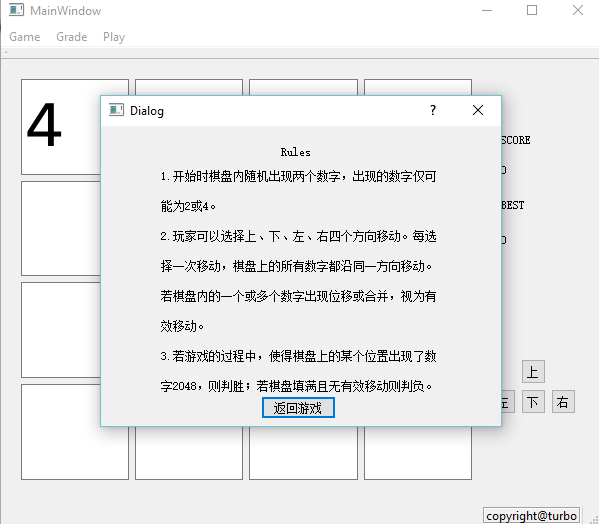


图2.5 规则提示窗口弹出

其二，该系统通过在不同界面的特定功能控制文件的多次读写。该系统从玩家注册账户到完成一次游戏一共完成了两次文件的读写过程。在注册界面，如图2.6，当玩家在输入框中输入自己的用户名后，点击“注册游戏”，系统就会开始读取源程序目录下的文件“info.txt”，获取文件中的信息，如图2.7；文件中存储的是所有玩家的游戏信息，包括用户名、排名、最好成绩，如图2.8，需要注意的是当玩家没有完成过游戏的情况下他的排名始终为0；当用户名输入符合要求后就会将信息重新写入该文件中。这些功能都是通过“注册用户”绑定的槽函数实现的，所以控制注册界面的“注册账户”按钮就实现了对文件读写的一次控制。在登录界面，如图2.9，当玩家在输入框中输入自己的用户名后，点击“登录游戏”，系统就会开始读取源程序目录下的文件“info.txt”，获取文件中的信息，如图2.7；当用户名输入符合要求后会跳转到游戏界面，如图2.10；当玩家结束游戏后，系统就会将更新后的数据重新写入文件中；也就是说，登陆界面中的按钮“登陆游戏”和游戏界面游戏结束标志就是系统另一次控制文件的读写。

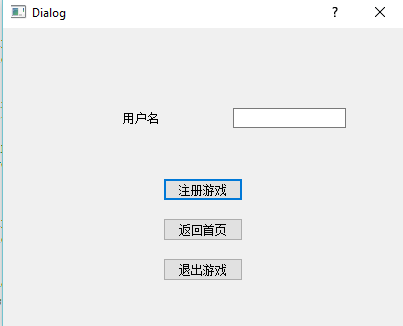


图2.6 注册界面

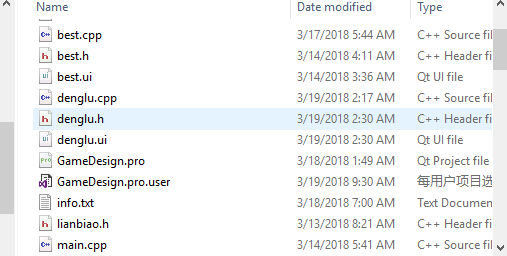


图2.7 “info.txt”所在目录

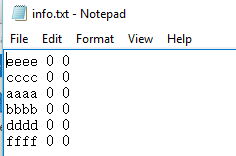


图2.8 “info.txt”文件内容

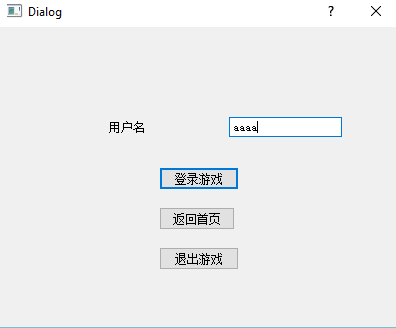


图2.9 登录界面

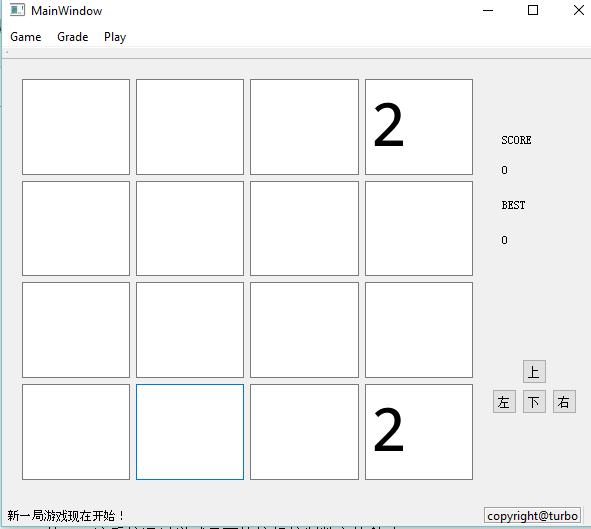


图2.10 游戏界面

其三，该系统通过游戏界面的按钮控制数字的移动。在游戏界面的右下角处有四个方向按钮，如图2.11；玩家可以通过点击其中一个按钮来移动表格中的数字，点击按钮是一个信号，发送至对应的槽函数，槽函数接受信号之后会开始运行；比如按钮“上”，数字就会向上移动。也就是说，系统通过控制按钮的触发来控制按钮背后的槽函数的运行，如图2.12为触发按钮之前的状态，当点击按钮“上”后，界面如图2.13所示。



图2.11 方向按钮

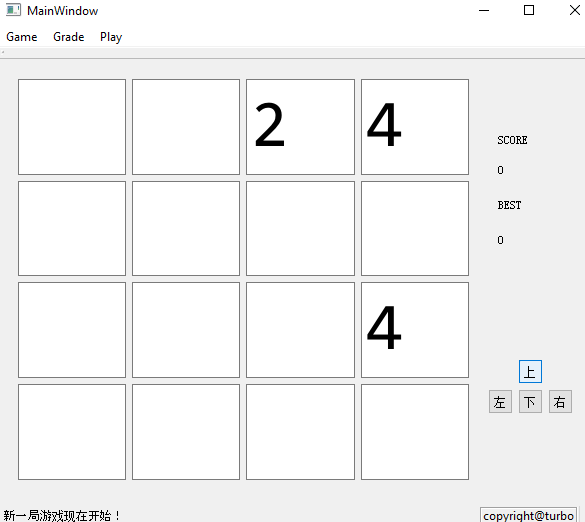


图2.12 执行“上”操作前

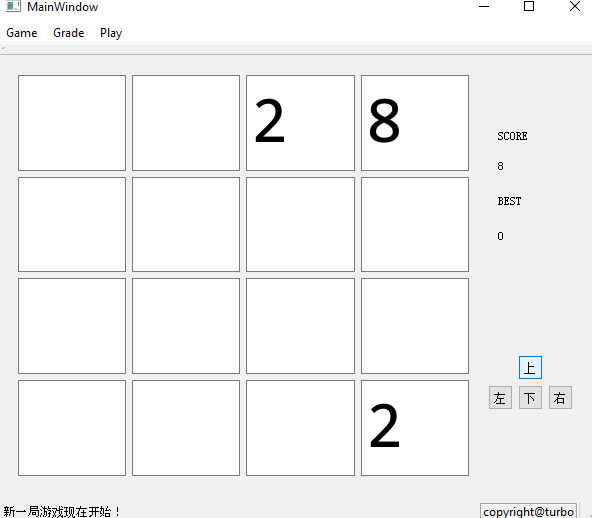


图2.13 执行“上”操作后

由上述三个方面可以发现，该系统最大的控制特点就是通过将按钮和对应的槽函数绑定，只有在系统读取到特定按钮发送的信号才会执行相应的函数，从而实现玩家可以通过控制按钮实现对游戏进程的控制。

其四，该系统通过十字链表控制游戏过程中数据的变化和存储。

对于该游戏运行后从文件“info.txt”中读取的数据将使用一个新创建的十字链表进行临时存储，并通过对链表的操作实现对数据的更新。该链表分为两个部分，分别是横向链表和纵向链表，结构如图2.14所示。横向链表的结构定义如图2.15，该链表中的结点的数据域存储的是每个玩家的基本信息，包括该玩家的用户名、最好成绩、最好成绩排名；指针域有两个指针，其一指向下一个玩家，其二指向存储该玩家该局游戏进程的链表，也就是纵向链表。纵向链表的结构定义如图2.16，每个玩家对应的纵向链表中结点的数据域存储的是该玩家在正在进行的该局游戏中每次执行操作后的状态，也就是表格中的数据以及执行完这步操作后的分数；该链表是一个双向链表，前指针指向执行该操作前一个状态，从而高效率地实现悔棋操作；需要注意的是，每次游戏结束后纵向链表会清空，每次游戏开始重新创建。游戏进行过程中，每执行完一步运动操作就会在判断游戏是否结束，如果没有结束就会创建一个新的结点，使用这个新的结点执行下一步操作。也就是说，随着游戏的进行，十字链表中的数据会实时更新，系统就是通过操作十字链表实现对游戏进程中的数据变化的存储。

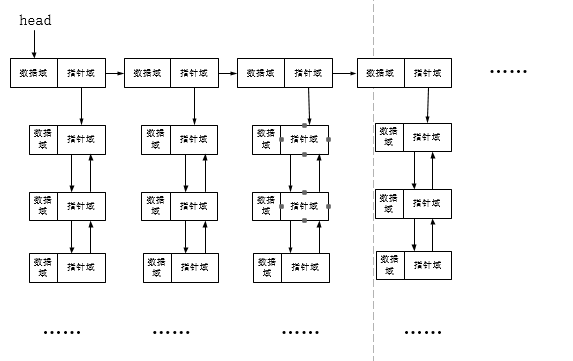


图2.14 十字链表结构

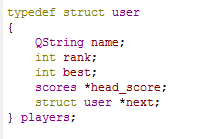


图2.15 横向链表定义

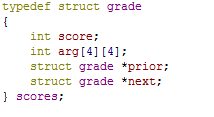


图2.16 纵向链表定义

## 2.2 系统实现的原理

该系统分为多个模块，系统通过对各模块的功能实现从而实现整个系统的功能。

该系统一共分为三大模块，结构如图2.17，分别是提示模块、账户管理模块以及游戏操作模块。

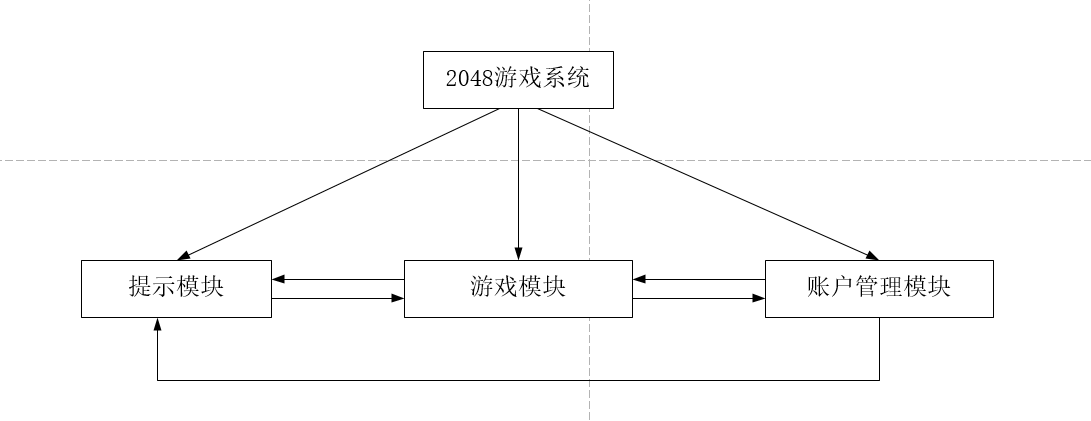


图2.17 系统结构图

提示模块，其结构如图2.18，负责游戏进程中玩家的操作出现错误或者不符合要求时运行的模块，主要实现的功能有提示玩家其输入是否符合要求、提示玩家游戏规则、提示玩家其个人信息以及显示游戏进程等。

游戏模块，其结构如图2.19，玩家进行游戏过程中主要实现的模块，主要实现包括玩家游戏操作的选择、玩家选择的操作对应玩家操作结果的显示以及玩家成绩等信息的更新等在内的功能。

账户管理模块，也是账户信息管理模块，其结构如图2.20，负责对所有玩家的游戏进行管理和基本操作的模块，最主要的功能就是对玩家信息文件“info.txt”的读写，读取文件中信息后存储到新建的链表中进行管理，游戏结束后将链表中更新后的信息重新写入原文件中，实现对玩家账户信息在游戏过程中的读写管理；该模块还会对横向链表中的数据进行操作实现对玩家最好成绩重新排名等。

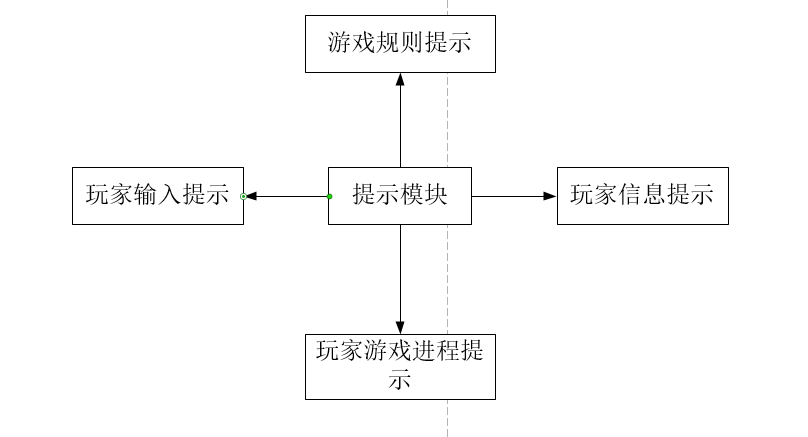


图2.18 提示模块结构

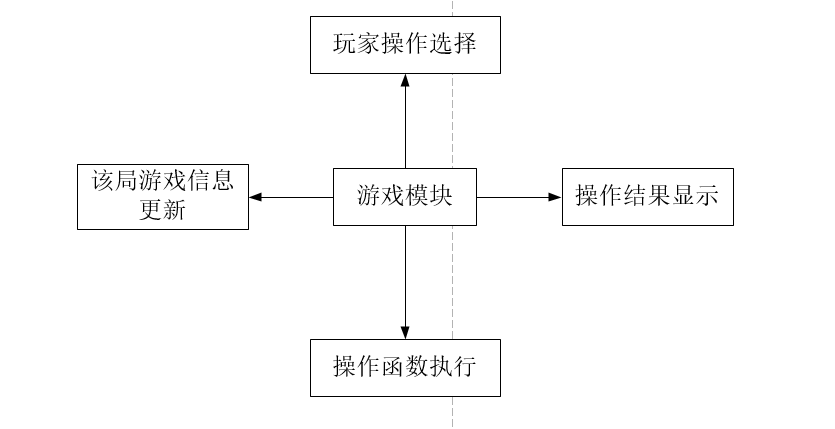


图1.19 游戏模块结构

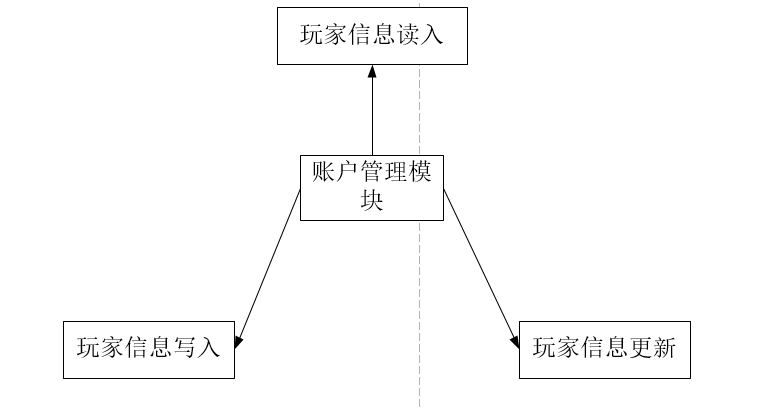


图1.20 账户管理模块结构

根据图2.17，可以发现，三个模块不是完全相互独立的，游戏模块中特定条件触发，就会运行提示模块中的程序，同时提示模块中特定条件触发也可以重新回到游戏模块继续游戏进程；同时有的提示模块实现过程中需要调用账户管理模块中的程序获取账户信息实现信息的提示功能。

## 2.3 系统实现方案分析

该2048游戏系统实现方案中，将系统需要实现的功能进行模块化，这样不仅便于检测系统功能的实现，而且可以大大降低系统出错后修改的难度和时间耗费。

首先，对系统模块化进行分析；合理的系统模块化后，各个模块

# 3 系统各模块的设计

## 3.1 系统各模块的界面设计

提示模块，主要由tip界面、tips界面、best界面、rank界面、rule界面、win界面以及over界面组成。

tip界面，ui设计的文件名为tip.ui，如图3.1，界面中的提示信息使用的是label控件，内容为“该用户不存在”；下方有个pushbutton控件，用于关闭窗口回到游戏界面；玩家在登录界面输入用户名后，点击“登录游戏”，若文件中不存在该用户，就会弹出tip窗口，提示玩家。



图3.1 tip界面

tips界面，ui设计的文件名为tips.ui，如图3.2，界面中的提示信息使用的是label控件，内容为“该用户名已存在”，下方有个pushbutton控件，用于关闭窗口回到游戏界面；玩家在注册界面输入用户名后，点击“注册游戏”，若文件中存在该用户，就会弹出tips窗口，提示玩家。



图3.2 tips界面

best界面，ui设计的文件名为best.ui，如图3.3，界面中的信息由两部分组成，分别是该玩家的用户名和最好成绩，内容的显示是通过label控件完成的；下方有个pushbutton控件，用于关闭窗口回到游戏界面。在游戏界面的菜单栏，玩家选择“Best”菜单项就会弹出best窗口告知玩家有关信息。



图3.3 best界面

rank界面，ui设计的文件名为rank.ui，如图3.4，界面中的信息同样由两部分组成，分别是该玩家的用户名和最好成绩排名，玩家初始最好成绩排名为0；下方有个pushbutton控件，用于关闭窗口回到游戏界面；在游戏界面的菜单栏，玩家选择“rank”菜单项就会弹出rank窗口告知玩家有关信息。

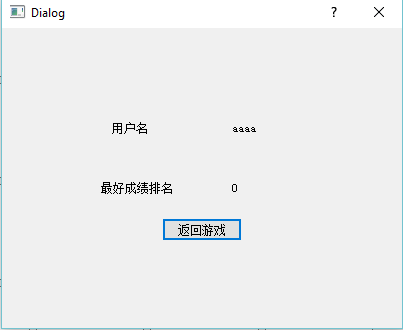


图3.4 rank界面

over界面，ui设计的文件名为over.ui，如图3.5，界面中的提示信息为“Game Over!”；玩家在游戏失败或者选择游戏界面菜单栏中菜单项“End Game”，就会跳转到over界面提示玩家游戏失败。在界面中有两个按钮，玩家可以选择不同按钮实现不同功能。



图3.5 over界面

rule界面，ui设计的文件名为rule.ui，如图3.6，界面中的提示信息为“Rules…”；使用了多个label控件显示信息，下方有个pushbutton控件，用于关闭窗口回到游戏界面；玩家在游戏界面的菜单栏中菜单项“Rules”，就会弹出rule界面提示玩家游戏规则。

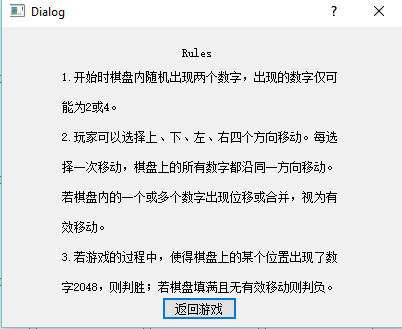


图3.6 rule界面

win界面，ui设计的文件名为win.ui，如图3.7，界面中的提示信息使用label控件，内容为“游戏成功”；在窗口下方有两个pushbutton控件，玩家选择不同的按钮实现返回登录界面和退出游戏的功能；玩家游戏成功后就会跳转到win窗口告知游戏成功。



图3.7 win界面

游戏模块，由游戏界面也就是主界面组成。

mainwindow界面，ui设计的文件名为mainwindow.ui，如图3.8，界面上方为游戏的菜单栏，有三个一级菜单项，不同一级菜单项下有不同的二级菜单项，有New Game、End Game、Rules、Rank、Best以及Back选项，点击不同菜单项可以实现相应的功能；界面中央的16个方格使用Grid Layout控件和textEdit控件设计而成的，然后对textEdit控件禁用编辑，使得玩家对方格不能进行编辑，从而用于显示游戏中移动后的数字结果；界面右下角的“上、下、左、右”四个pushbutton控件作为玩家选择数字移动方向的按钮，右上方的四个label控件用于显示游戏进程中的相关信息。

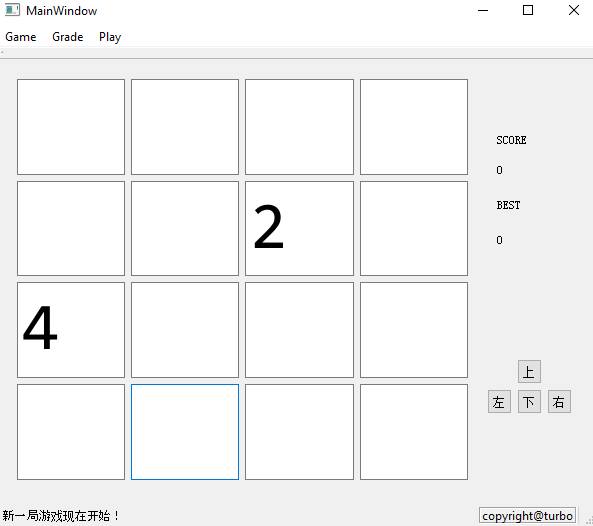


图3.8 mainwindow界面

账户管理模块，由denglu界面、zhuce界面组成。

dneglu界面，ui设计的文件名为denglu.ui，如图3.9，中央有一个label控件显示“用户名”，提示玩家在右边的输入框中输入玩家自己的用户名，该输入框是一个lineEdit控件，可以进行编辑；下方是三个pushbutton控件，玩家选择不同的按钮可以实现登录游戏、返回首页以及退出游戏的功能。

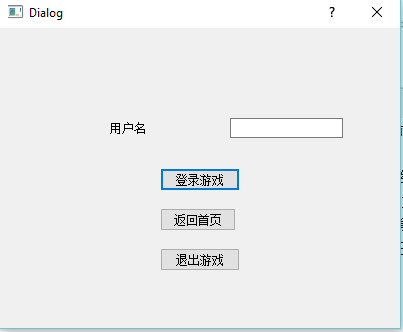


图3.9 denglu界面

zhuce界面，ui文件设计的文件名为zhuce.ui，如图3.10，和denglu界面很相似的就是，中央同样是一个label控件和lineEdit控件，下方同样是三个pushbutton控件，但是实现的是注册账户、返回首页以及退出游戏的功能。



图3.10 zhuce界面

上述就是该系统各个模块的界面设计，界面设计是使用QT的design功能完成的，这样大大降低了设计难度，也让各个界面之间的连接直接通过设计过程中设计的槽函数实现了。

## 3.2 系统各模块的功能设计

提示模块，该模块主要功能是提示玩家其想要获取的相关信息；现在就提示模块的功能设计进行分析。

首先根据玩家进行游戏需要获取的内容将模块进行进一步细分，那么玩家进行游戏过程中需要获取哪些信息呢？首先是玩家的个人信息，玩家的个人信息包括该玩家的最好成绩、最好成绩排名；其次是游戏中有关游戏进程的提示，游戏进程提示包括游戏失败、游戏成功；还有就是游戏开始时玩家账户信息提示，账户信息包括注册时输入用户名已经存在和登录时用户名不存在；最后就是游戏相关信息，比如规则。

根据上述对提示内容的细分，如图3.11,我们将在该模块中设计了多个界面从而实现对这些内容的显示，如图3.12，每个界面除了显示这些信息的功能外还有跳转界面等功能。

tip界面（如图3.1）、tips界面（如图3.2）、rank界面（如图3.4）、best界面（如图3.3）以及rule界面（如图3.6）是最简单的信息提示界面，功能就只有提示用户，只是提示的内容不一样而已。但是他们的触发条件不一样，tip界面、tips界面是玩家在注册、登录游戏时输入用户名不正确时弹出的，所以是自发性弹出；但是其他界面是玩家在选择相应菜单项后弹出的界面，属于被动性弹出。这些界面下方的按钮“返回游戏”，玩家点击这个按钮后就可以关闭该窗口，重新回到原来的界面继续游戏。

win界面（如图3.7）以及over界面（如图3.5）中除了label控件提示用户游戏成功以及游戏失败外，pushbutton按钮被点击后也可以实现特定功能。两个界面中共有的按钮“退出游戏”被点击后就会关闭游戏，win界面的“返回登录界面”被点击后就会跳转到登录界面，让玩家重新登录游戏，over界面的“重新开始游戏”按钮被点击后就会重新打开一个新的游戏界面，玩家继续在该用户名下继续新的一局游戏；这就是这两个界面实现的其他功能。

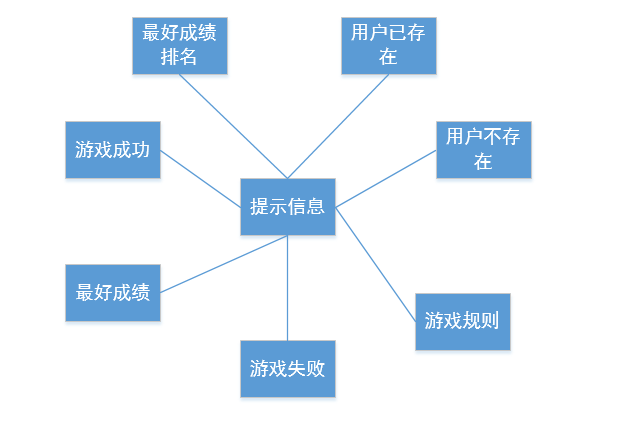


图3.11 提示信息内容

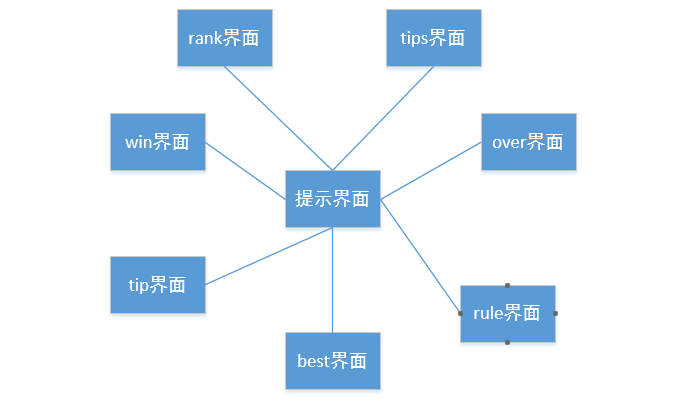


图3.12 提示模块界面设计组成

游戏模块，该模块的主要功能就是供玩家进行游戏操作并显示每步操作结束后的状态显示。

首先，我们需要通过对游戏规则的理解获取游戏有关的操作类型，我们可以发现玩家需要进行的操作就是上下左右移动数字，直到游戏结束；其次，每进行一次操作，方格中的数字就会更新，分数也会累加；最后，每进行一次游戏操作，就需要对游戏是否结束进行一次判定，如果游戏失败则执行某个函数，若游戏成功，则执行某个函数。

对于玩家可能进行的操作，我们在游戏界面（如图3.8）设计了四个按钮，他们的功能之一就是被点击后就执行与其绑定的函数，实现数字的移动，界面显示的是移动后的结果。正如上面说的那样，每次操作都会对分数进行更新和游戏结束的判断，所以按钮绑定的函数同样实现了分数的累加和游戏是否结束的判断。如果游戏结束就会弹出相应的提示窗口，这样就跳转到提示模块的函数运行了。

账户管理模块，该模块的主要功能是管理玩家注册的账户。

首先，对于玩家的信息，每次游戏过程中都需要使用，尤其每个玩家的最好成绩、用户名等；其次，为了保存玩家的信息，我们需要创建一个文件如“info.txt”来存储这些信息，每次注册完毕和游戏结束的时候写入文件，每次注册和开始游戏的时候读取文件中的内容并存入到一定空间中。

所以在管理模块，我们需要完成对文件的读写，而读写功能在登录界面（如图3.9）、注册界面（如图3.10）以及主界面完成（如图3.8）。

在注册界面，玩家输入用户名后，系统开始读取文件中的信息，将输入的信息与读取出来的信息进行比对，如果符合要求则将更新后的信息写入原来的文件中，并跳转到登录界面；在登录界面，玩家输入用户名后，系统开始读取文件中的信息，将输入的信息与读取出来的信息进行比对，如果符合要求则将更新后的信息写入原来的文件中，并跳转到游戏界面。那么，文件中的信息怎么存储到一定空间呢？每次注册或者登录，我们都会创建一个新的十字链表，将信息按顺序存入横向链表中，这样就完成了读取文件中信息，比较过程只需要遍历链表就可以实现；写入文件就只需要将链表中的内容按一定顺序写入原来的文件中。

# 4 系统各模块的功能实现

## 4.1 系统各模块功能的算法设计

该系统的总流程图如图4.1所示。

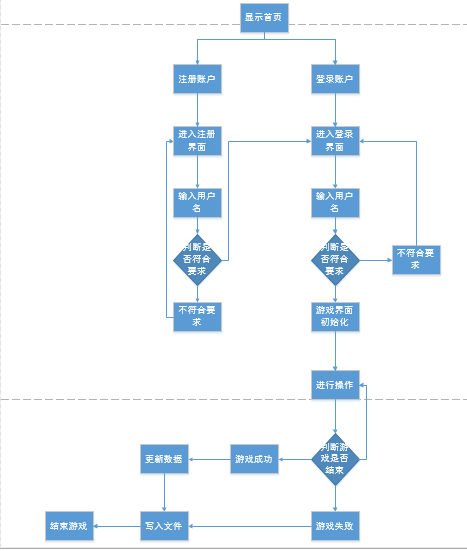


图4.1 系统流程图

根据上面章节对各模块功能的设计，在本章节将对各功能的设计的算法部分进行分析，同样是按照模块对功能进行分析。

### 4.1.1 提示模块

提示模块的主要功能有显示信息、关闭窗口、跳转界面等。

对于tip界面（如图3.1）、tips界面（如图3.2）以及rule界面（如图3.6）；使用Qt的design功能，将显示的信息写到对应的label控件中，当窗口弹出时，信息就会在窗口确定的位置显示；使用Qt的design功能，将一个pushbutton控件放在界面中，再对该控件绑定一个槽函数，槽函数中实现关闭窗口，这样只需要点击该按钮就可以关闭该窗口回到原来的界面。槽函数中的关闭窗口的函数运用的是Qt库中自带的函数close()，如图4.2。

对于rank界面（如图）以及best界面（如图）；使用Qt的design功能，将不需要调用文件中内容的显示信息写到对应的label控件中，当窗口弹出时，信息就会在窗口确定的位置显示，对于该玩家的最好成绩排名和最好成绩，系统会从该玩家对应的横向链表中的结点的数据域中获取；使用Qt的design功能，将一个pushbutton控件放在界面中，再对该控件绑定一个槽函数，槽函数中实现关闭窗口，这样只需要点击该按钮就可以关闭该窗口回到原来的界面。槽函数中的关闭窗口的函数运用的是Qt库中自带的函数close()，如图4.2。



图4.2 关闭窗口函数

对于win界面（如图）以及over界面（如图），提示信息的显示方式和上述是一样的；但不同的是这两个界面还有跳转界面的功能。在Qt下跳转界面同样很容易实现，只需要创建一个新的所需跳转界面的类型的界面，然后关闭该界面就可以了。如图4.3为注册界面跳转到登录界面的函数，this就是当前窗口的类。



图4.3 创建新界面函数

### 4.1.2 游戏模块

游戏模块的功能都是在主界面实现的。

对于数字的移动，主要分两部分进行；第一部分是将所有的数字按照选择的方向移动到不能进行移动，第二部分就是数字累加过后按照同样的方向进行移动。

现对移动函数的算法进行设计和分析。

以向上移动为例，如图4.4为向上移动的函数源程序，函数名为“move\_up”；在移动函数中，从左到右一列一列的进行移动；在每一列的移动中，首先判断i和i+1行是否都不为0,若都不为0，则这两行数字不需要移动；若i行数字为0，则需要将i+1行的数字赋给前一行，再将后一行的数字赋值0；若全为0，则对i+2行数字是否为0进行判断，若不为0，则将i+2行的数字赋给i行，i+2行赋值0；若为0，则对i+3行的数字进行判断，若为0则不移动，若不为0，则将i+3行的数字赋给i行的数字，同时i+3行赋值0。这些判断过程中，都需要对i+1、i+2和i+3的值是否符合要求进行判断。



图4.4 move\_up

对移动后的数字进行累加，每次移动只需要进行一次。

现对数字累加函数进行算法设计和分析。

以向上移动后的累加函数为例，如图4.5为以向上移动后的累加函数的源程序，函数名为“add\_up”；在累加函数中，从左至右一列一列的进行操作，需要判断的是i行数值和i+1行的数值是否相等，不相等则不想加，相等则累加，然后对分数进行更新。

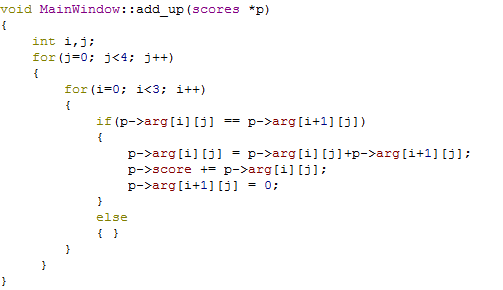


图4.5 add\_up

对于移动累加移动后的数字显示，运用Qt的design功能，在源程序中调用Qt库中的setText函数和setFont函数来显示数字和设置数字字体大小。但是因为对于数字为0时不显示的情况，所以显示每个数字前需要对这个数字的值进行判断，若为0则显示空字符就是没有显示；若不为0，则照旧显示该数字。

对于每进行一次操作后分数的显示，运用Qt的design功能，然后和前面提示模块一样，获取每进行一次操作后状态的指针的数据域的分数，只需要将其显示出来就可以了。

在该模块还有一个最大的功能，也就是每次操作结束后的判断和下次操作进行前的准备。

对于这个功能，源程序中设计了一个函数实现，也就是“GameOver”函数，如图4.6所示，是该函数的流程图。



图4.6 GameOver函数流程图

对于图4.6中的创建新节点，具体算法如图4.7。

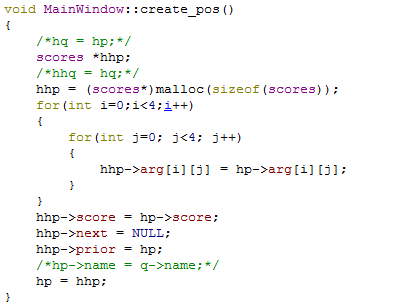


图4.7

## 4.2 系统各模块功能的运行测试

运行程序。

1.进入首页，如图4.8.



图4.8

2.点击“注册”，进入注册界面，如图4.9。



图4.9

3.点击“返回首页”，返回到首页，如图4.10



图4.10

4.点击“注册”，再次来到注册界面，并输入用户名“gggg”，如图4.11



图4.11

5.点击“注册游戏”，跳转到登录界面，如图4.12，打开文件“info.txt”，可以看到注册的账户“gggg”的信息，如图4.13.



图4.12

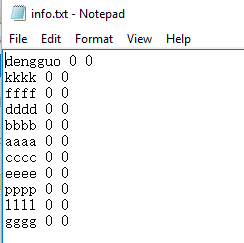


图4.13

6.在登录框中输入“gggg”，如图4.14

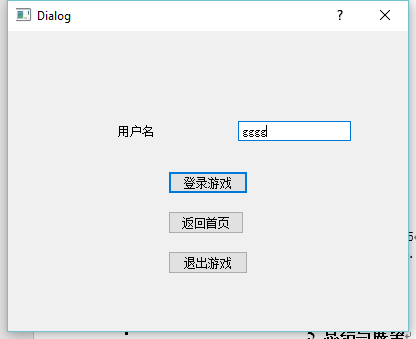


图4.14

7.点击“登录”，跳转到游戏界面，如图4.15

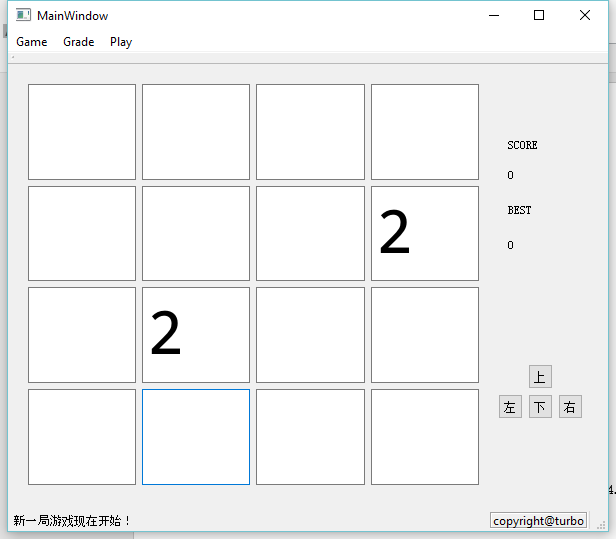


图4.15

8.依次对数字进行上、左、上、右、下、左、下操作，每次操作结果如图4.16、4.17、4.18、4.19、4.20、4.21以及4.22.

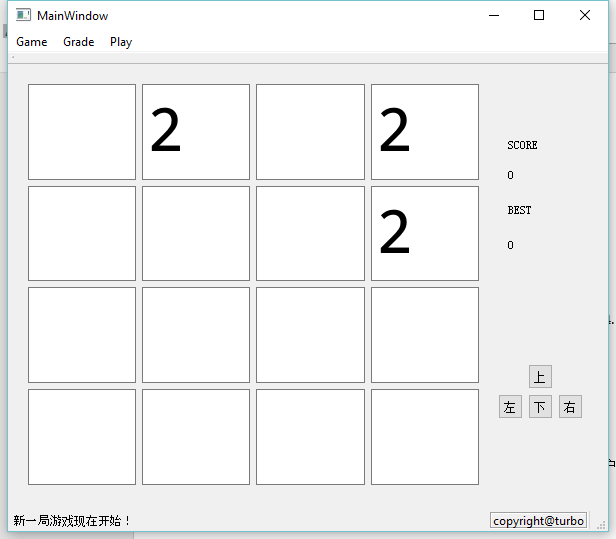


图4.16

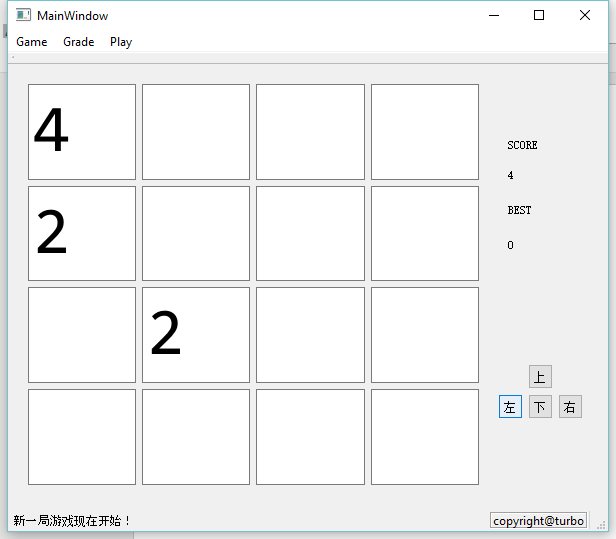


图4.17

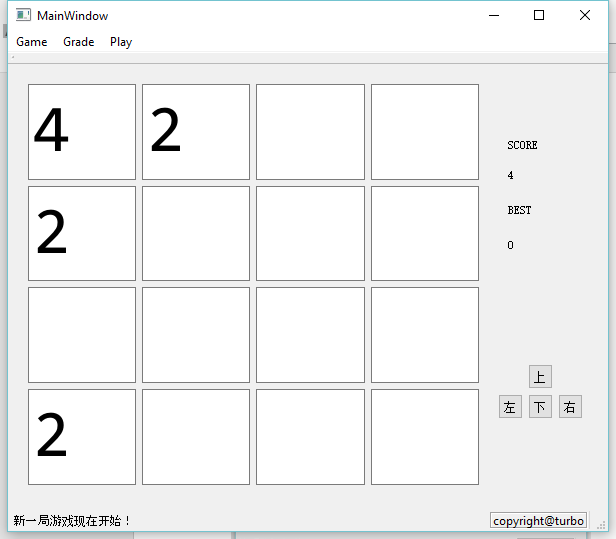


图4.18

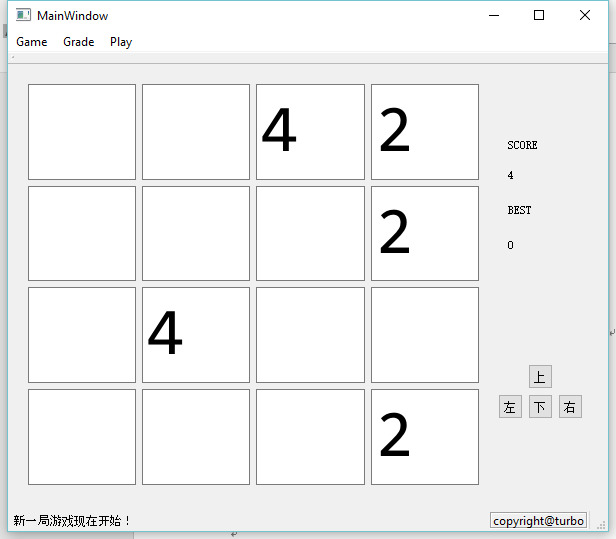


图4.19

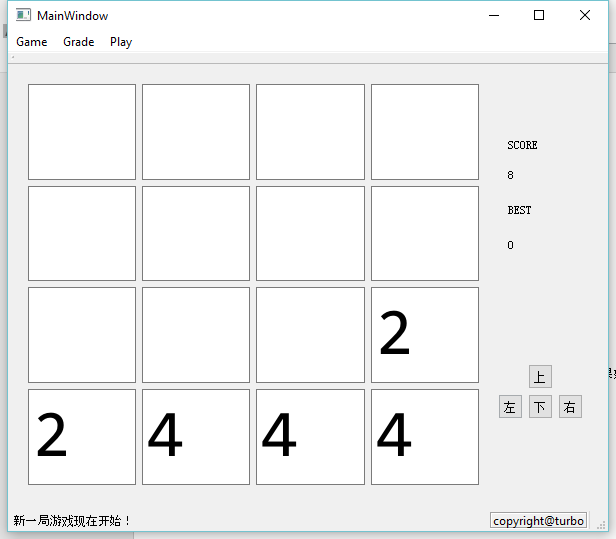


图4.20

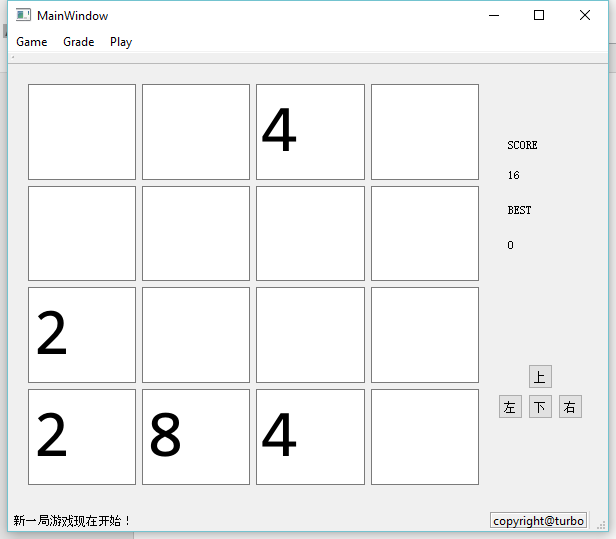


图4.21

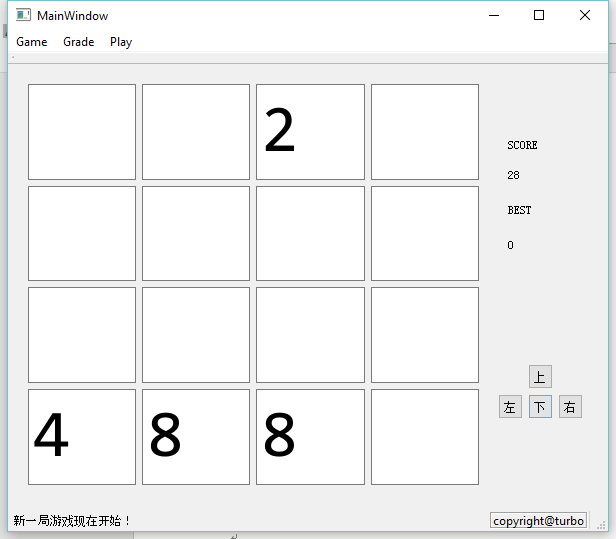


图4.22

9.进行悔棋操作，执行结果依次如图4.23、4.24、4.25、4.26、4.27、4.28以及4.29.

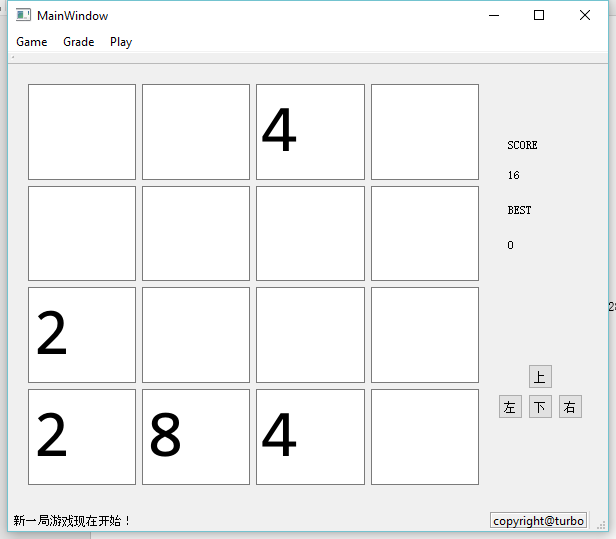


图4.23

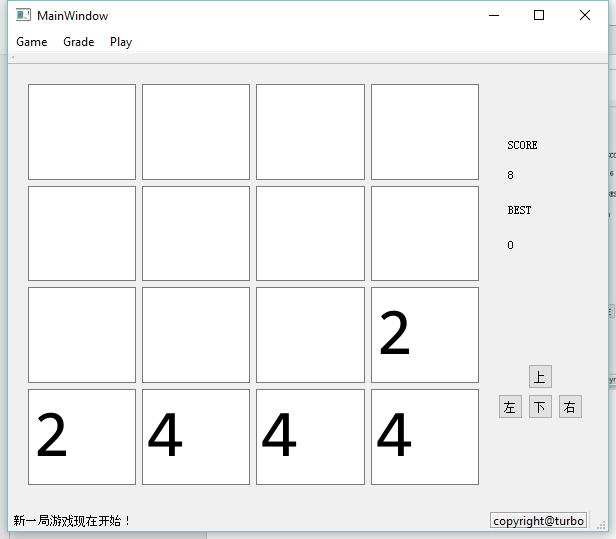


图4.24

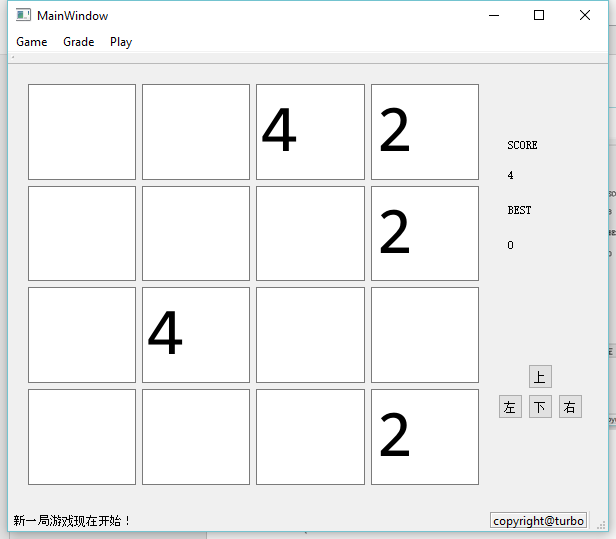


图4.25

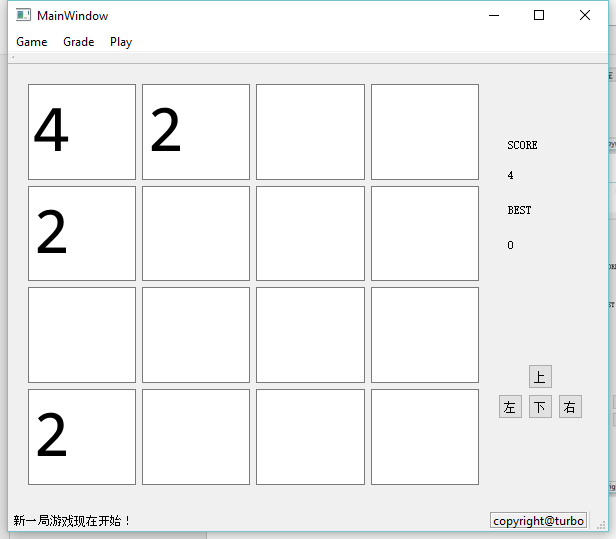


图4.26

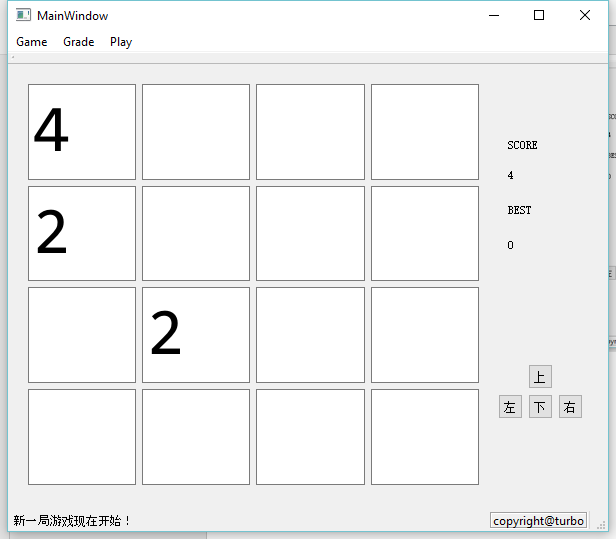


图4.27

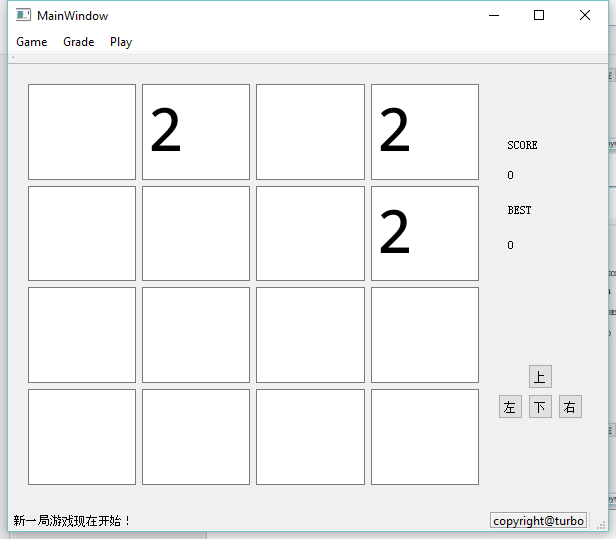


图4.28

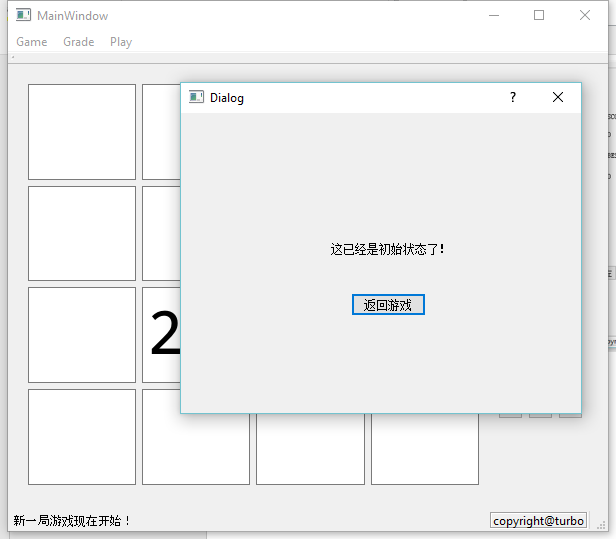


图4.29

10.继续下棋，一次操作上、下、左、右，操作结果如图4.30、4.31、4.32、4.33。

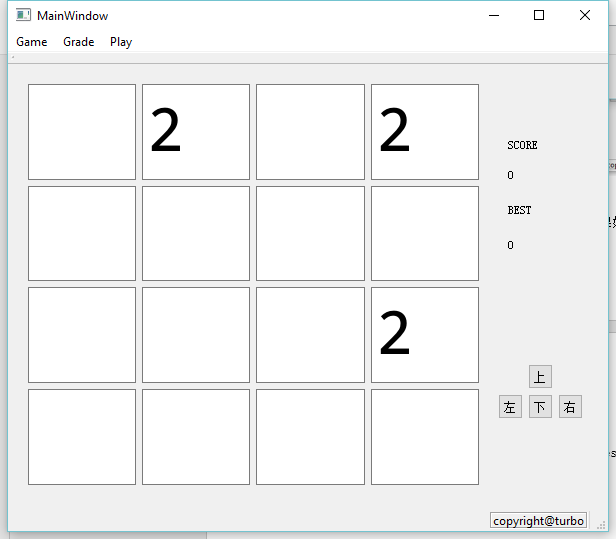


图4.30

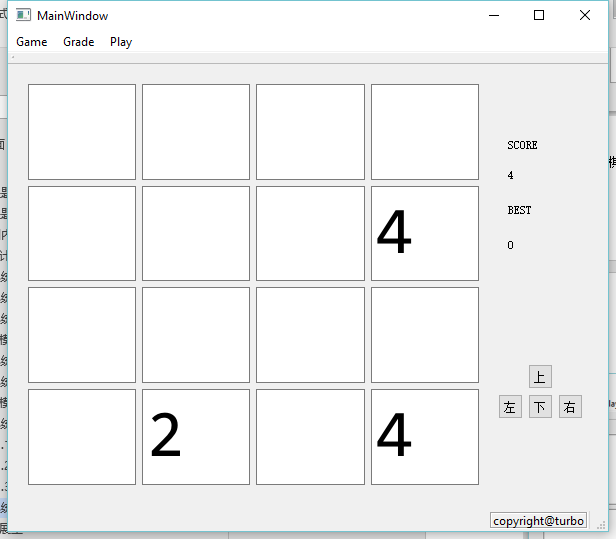


图4.31

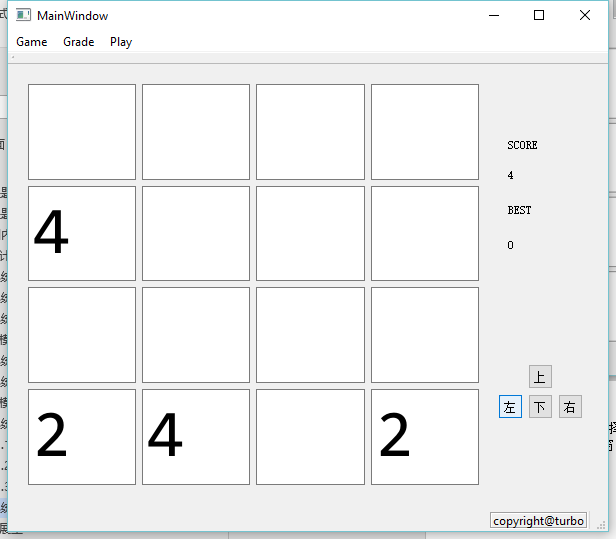


图4.32

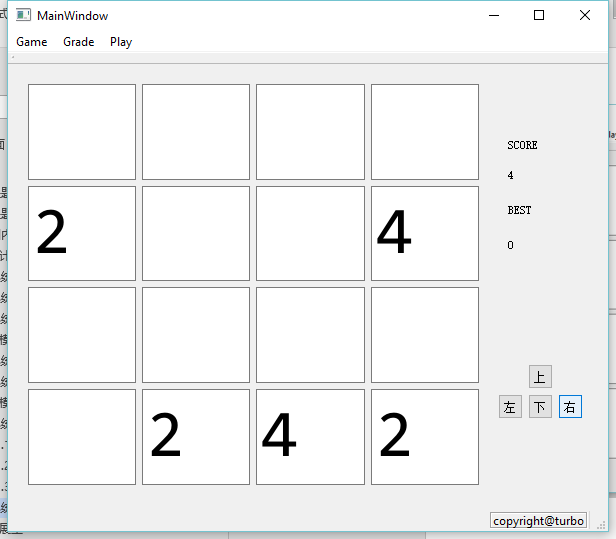


图4.33

11.依次选择菜单项“Rank”、“best”以及“rules”，依次弹出排名、最好成绩以及规则窗口；如图4.34、4.35、4.36。



图4.34



图4.35

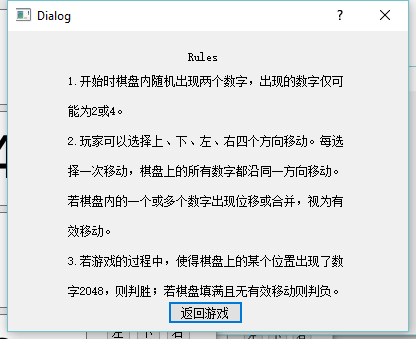


图4.36

12.菜单项选择“New Game”，打开一个新的游戏界面，如图4.37；

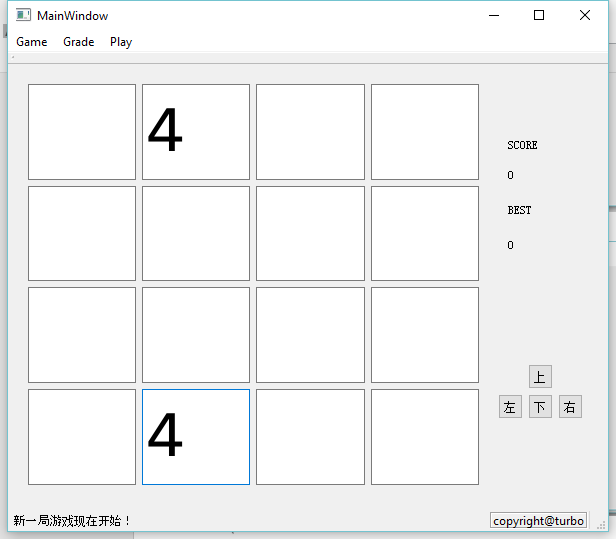


图4.37

13.菜单项选择“Ene Game”，则跳转到结束界面，如图4.38；



图4.38

14.选择“重新开始游戏”，则打开一个新的游戏界面，如图4.39

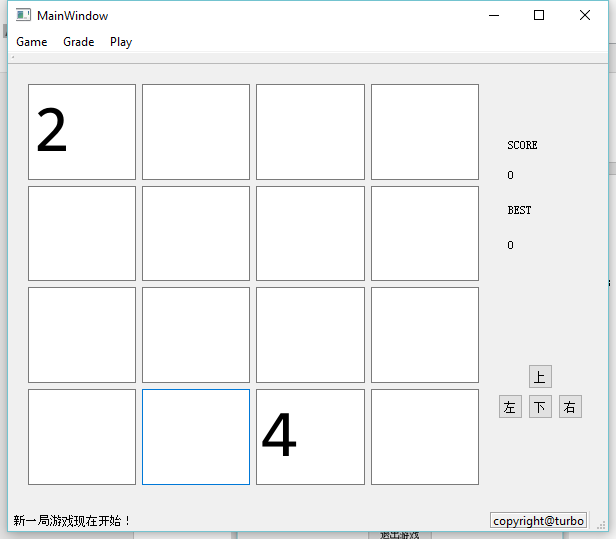


图4.39

15. 菜单项选择“Ene Game”，则跳转到结束界面，如图4.40；

、

图4.40

16.选择“退出游戏”，则窗口关闭。

# 5 总结与展望

本次课程设计任务是在初始的2048游戏的基础上添加玩家账户管理系统和悔棋功能。

对于任务中要求的图形界面的设计，根据我对已经掌握的知识的回顾，果断选择了软件Qt，Qt是一个跨平台[C++](https://baike.baidu.com/item/C%2B%2B)[图形用户界面](https://baike.baidu.com/item/%E5%9B%BE%E5%BD%A2%E7%94%A8%E6%88%B7%E7%95%8C%E9%9D%A2)应用程序开发框架；它既可以开发[GUI](https://baike.baidu.com/item/GUI)程序，也可用于开发非GUI程序，比如控制台工具和服务器。虽然Qt对于图形界面的设计可以大大降低设计者的代码量的同时更加美观，但是对我来说依然是一款陌生的开发平台。在我高涨的研究热情的下，这个平台依然成为我的选择，尽管在设计过程中遇到过许多困难依然没有选择放弃；反而在不断优化算法和修改错误的过程中学习和掌握了更多知识和技能，这也是老师不断提到的在实践中对自己本来已经学习的知识进行运用并在其进行完善和升华；我在课设过程中，不仅对我掌握的代码知识和数据结构进行了运用，而且在运用过程中学到了更多的知识和技能，最大的收获莫过于从Qt小白到能够运用Qt完成一个小游戏的设计的设计者。

# 参考文献

[1]曹计昌,卢萍,李开. C语言与程序设计. 北京：电子工业出版社,2013.1

[2]李开,卢萍,曹计昌. C语言实验与课程设计.北京：科学出版社,2011.3

[3]霍亚飞. Qt Creator 快速入门. 北京：北京航空航天大学出版社，2017.1

[4]金大. Qt 5 开发实践. 北京：人民邮电出版社，2015.9

[5]严蔚敏，吴伟民. 数据结构. 北京：清华大学出版社 2004.2

# 附录

1. 基础部分源程序

best.h //最好成绩界面的头文件

#ifndef BEST\_H

#define BEST\_H

#include <QDialog>

#include "lianbiao.h"

namespace Ui {

class Best;

}

class Best : public QDialog

{

Q\_OBJECT

public:

explicit Best(QWidget \*parent = 0);

~*Best*();

private slots:

void on\_pushButton\_clicked();

private:

Ui::Best \*ui;

};

#endif // BEST\_H

denglu.h //登录界面头文件

#ifndef DENGLU\_H

#define DENGLU\_H

#include <QDialog>

#include <QVBoxLayout>

#include <QFileInfo>

#include <QTextCodec>

#include <QString>

#include <QFile>

#include <QDebug>

#include "lianbiao.h"

/\*extern players \*playr;\*/

namespace Ui {

class Denglu;

}

class Denglu : public QDialog

{

Q\_OBJECT

public:

explicit Denglu(QWidget \*parent);

~*Denglu*();

private slots:

void on\_pushButton\_2\_clicked();

void on\_pushButton\_clicked();

void on\_pushButton\_3\_clicked();

private:

Ui::Denglu \*ui;

void readInfo(QString line);

};

#endif // DENGLU\_H

lianbiao.h //构建的链表所在头文件

#ifndef LIANBIAO

#define LIANBIAO

#include <stdio.h>

#include <stdlib.h>

#include <QString>

typedef struct grade

{

int score;

int arg[4][4];

struct grade \*prior;

struct grade \*next;

} scores;

typedef struct user

{

QString name;

int rank;

int best;

scores \*head\_score;

struct user \*next;

} players;

#endif // LIANBIAO

over.h //游戏失败提示界面头文件

#ifndef OVER\_H

#define OVER\_H

#include <QDialog>

#include "lianbiao.h"

namespace Ui {

class Over;

}

class Over : public QDialog

{

Q\_OBJECT

public:

explicit Over(QWidget \*parent = 0);

~*Over*();

private slots:

void on\_pushButton\_clicked();

void on\_pushButton\_2\_clicked();

private:

Ui::Over \*ui;

};

#endif // OVER\_H

rank.h //用户最好成绩排名界面头文件

#ifndef RANK\_H

#define RANK\_H

#include <QDialog>

#include "lianbiao.h"

namespace Ui {

class Rank;

}

class Rank : public QDialog

{

Q\_OBJECT

public:

explicit Rank(QWidget \*parent = 0);

~*Rank*();

private slots:

void on\_pushButton\_clicked();

private:

Ui::Rank \*ui;

};

#endif // RANK\_H

rule.h //规则窗口头文件

#ifndef RULE\_H

#define RULE\_H

#include <QDialog>

#include "lianbiao.h"

namespace Ui {

class Rule;

}

class Rule : public QDialog

{

Q\_OBJECT

public:

explicit Rule(QWidget \*parent = 0);

~*Rule*();

private slots:

void on\_pushButton\_clicked();

private:

Ui::Rule \*ui;

};

#endif // RULE\_H

shouye.h //首页头文件

#ifndef SHOUYE\_H

#define SHOUYE\_H

#include <QDialog>

#include "lianbiao.h"

namespace Ui {

class Shouye;

}

class Shouye : public QDialog

{

Q\_OBJECT

public:

explicit Shouye(QWidget \*parent = 0);

~*Shouye*();

private slots:

void on\_pushButton\_2\_clicked();

void on\_pushButton\_clicked();

private:

Ui::Shouye \*ui;

};

#endif // SHOUYE\_H

tip.h //用户不存在提示窗口

#ifndef TIP\_H

#define TIP\_H

#include <QDialog>

namespace Ui {

class Tip;

}

class Tip : public QDialog

{

Q\_OBJECT

public:

explicit Tip(QWidget \*parent = 0);

~*Tip*();

private slots:

void on\_pushButton\_clicked();

private:

Ui::Tip \*ui;

};

#endif // TIP\_H

tips.h //该用户名已存在提示窗口头文件

#ifndef TIPS\_H

#define TIPS\_H

#include <QDialog>

namespace Ui {

class Tips;

}

class Tips : public QDialog

{

Q\_OBJECT

public:

explicit Tips(QWidget \*parent = 0);

~*Tips*();

private slots:

void on\_pushButton\_clicked();

private:

Ui::Tips \*ui;

};

#endif // TIPS\_H

win.h //游戏成功后弹出窗口头文化

#ifndef WIN\_H

#define WIN\_H

#include <QDialog>

namespace Ui {

class Win;

}

class Win : public QDialog

{

Q\_OBJECT

public:

explicit Win(QWidget \*parent = 0);

~*Win*();

private slots:

void on\_pushButton\_clicked();

void on\_pushButton\_2\_clicked();

private:

Ui::Win \*ui;

};

#endif // WIN\_H

zhuce.h //注册界面头文件

#ifndef ZHUCE\_H

#define ZHUCE\_H

#include <QDialog>

#include <QCoreApplication>

#include <QVBoxLayout>

#include <QFileInfo>

#include <QTextCodec>

#include <QString>

#include <QFile>

#include <QDebug>

#include "lianbiao.h"

/\*extern players\* head;\*/

namespace Ui {

class Zhuce;

}

class Zhuce : public QDialog

{

Q\_OBJECT

public:

explicit Zhuce(QWidget \*parent = 0);

~*Zhuce*();

private slots:

void on\_pushButton\_clicked();

void on\_pushButton\_2\_clicked();

void on\_pushButton\_3\_clicked();

private:

Ui::Zhuce \*ui;

//读文件

void readInfo(QString line);

//写文件

void writeInfo(QTextStream &cin, players \*hhead);

};

#endif // ZHUCE\_H

mainwindow.h//游戏界面头文件

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QVBoxLayout>

#include <QFileInfo>

#include <QTextCodec>

#include <QKeyEvent>

#include <QString>

#include <QFile>

#include <QDebug>

#include "lianbiao.h"

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = 0);

~*MainWindow*();

private slots:

void on\_action\_New\_triggered();

//点击New Game

void on\_action\_End\_triggered();

//点击End Game

void on\_action\_Rules\_triggered();

//Rule

void on\_action\_Rank\_triggered();

//Rank

void on\_action\_Best\_triggered();

//Best

void on\_action\_Back\_triggered();

//Back

void on\_action\_Advance\_triggered();

//Advance

void on\_up\_clicked();

//up

void on\_left\_clicked();

//left

void on\_down\_clicked();

//down

void on\_right\_clicked();

//right

private:

Ui::MainWindow \*ui;

/\*void keyPressEvent(QKeyEvent \*event);\*/

//键盘触发函数

void Gameover(scores \*p, players \*q);

//每次移动后对游戏是否结束进行判断函数，并做出相应处理

void create\_pos(scores \*hhq);

//创建一个新的结点，先复制前一个状态结束时的状态

void move\_up(scores \*p);

//向上移动

void move\_down(scores \*p);

//向下移动

void move\_right(scores \*p);

//向右移动

void move\_left(scores \*p);

//向左移动

void add\_up(scores \*p);

//向上移动后的累加，包括分数的累加

void add\_down(scores \*p);

//向下移动后的累加

void add\_left(scores \*p);

//向左移动后的累加

void add\_right(scores \*p);

//向右移动后的累加

void rank\_new();

//更新排名

void show\_score(scores \*p);

//实时显示分数的函数

void show\_shuzi(scores \*p);

//实时显示表格中数字的函数

void show\_best(players \*pp);

//显示该用户最好成绩

void best\_update(players \*pp, scores \*p);

//对最好成绩进行更新的函数

void writeInfo(QTextStream &cin,players \*hhead);

//写入文件函数

};

#endif // MAINWINDOW\_H

best.cpp

#include "best.h"

#include "ui\_best.h"

extern players \*playr;

Best::Best(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Best)

{

ui->setupUi(this);

ui->label\_2->setText(playr->name);

ui->label\_4->setText(QString("%1").arg(playr->best));

}

Best::~*Best*()

{

delete ui;

}

void Best::on\_pushButton\_clicked()

{

this->close();

}

denglu.cpp

#include "denglu.h"

#include "ui\_denglu.h"

#include "denglu.h"

#include "shouye.h"

#include "mainwindow.h"

#include "tip.h"

players \*playr;

players \*head\_1;//头指针 在denglu界面

Denglu::Denglu(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Denglu)

{

ui->setupUi(this);

}

Denglu::~*Denglu*()

{

delete ui;

}

void Denglu::on\_pushButton\_2\_clicked()

{

this->close();

}

void Denglu::on\_pushButton\_clicked()

{

//读取文件

/\*int k = 0;\*/

int k1 = 0;

/\*head = (players\*)malloc(sizeof(players));\*/

head\_1 = new players;

head\_1->rank = 0;

head\_1->head\_score = NULL;

head\_1->name = "000";

head\_1->best = 0;

head\_1->next = NULL;

QFile file("G:/2048/GameDesign/info.txt");

if(!file.*open*(QIODevice::ReadOnly|QIODevice::Text))

return;

QTextStream in(&file);

/\*players \*pl;

pl = new players;\*/

while(!in.atEnd()){

QString line = in.readLine();

readInfo(line);//对读取的一行字符串进行处理

}

//获取信息

QString strText =ui->lineEdit->text();

//遍历用户信息

players \*p,\*q;

p = new players;

p->name = strText;

p->best = 0;

p->rank = 0;

p->head\_score = NULL;

for(q=head\_1->next;q->next!=NULL;)

{

if(QString::compare(q->name,p->name))//相等返回0

{

q->head\_score = NULL;

q = q->next;

/\*k = 0;\*/

}

else

{ //存在该用户

playr = q;

q = q->next;

k1 = 1;

}

}

//最后一个节点

q->head\_score = NULL;

if(q->next == NULL && (QString::compare(q->name,p->name)||k1))

{

if(QString::compare(q->name,p->name) && !k1)

{

//弹出提示界面

Tip \*tip = new Tip();

tip->show();

}

/\*k = 0;\*/

}

else

{

//存在该用户

playr = q;

k1 = 1;

}

while(k1)

{

MainWindow \*mainwindow = new MainWindow();

mainwindow->show();

this->close();

k1 = 0;

}

}

void Denglu::readInfo(QString line)

{

players \*p;

p = new players;

bool ok;

QStringList line\_1;

line\_1 = line.split(" ");

p->name = line\_1.at(0);

p->rank = line\_1.at(1).toInt(&ok,10);

p->best = line\_1.at(2).toInt(&ok,10);

p->next = head\_1->next;

head\_1->next = p;

}

void Denglu::on\_pushButton\_3\_clicked()

{

Shouye \*shouye = new Shouye;

shouye->show();

this->close();

}

main.cpp

#include "mainwindow.h"

#include <QApplication>

#include "shouye.h"

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

Shouye w;

w.show();

return a.exec();

}

mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "over.h"

#include "rule.h"

#include "win.h"

#include "best.h"

#include "rank.h"

#include "back.h"

#include <time.h>

scores \*hp, \*hq;

scores \*t;

extern players \*playr, \*head\_1;

int k;

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

srand(time(NULL));

ui->setupUi(this);

//状态栏显示 临时信息和永久信息

ui->statusBar->showMessage(tr("新一局游戏现在开始！"));

QLabel \* permanent = new QLabel(this);

permanent->setFrameStyle(QFrame::Box|QFrame::Sunken);

permanent->setText("copyright@turbo");

ui->statusBar->addPermanentWidget(permanent);

//连接槽函数

/\*connect(ui->action\_New, SIGNAL(triggered()),this,SLOT(on\_action\_New\_triggered()));

connect(ui->action\_End, SIGNAL(triggered()),this,SLOT(on\_action\_End\_triggered()));

connect(ui->action\_Rules,SIGNAL(triggered()),this,SLOT(on\_action\_Rules\_triggered()));

connect(ui->action\_Advance,SIGNAL(triggered()),this,SLOT(on\_action\_Advance\_triggered()));

connect(ui->action\_Back,SIGNAL(triggered()),this,SLOT(on\_action\_Back\_triggered()));

connect(ui->action\_Best,SIGNAL(triggered()),this,SLOT(on\_action\_Best\_triggered()));

connect(ui->action\_Rank,SIGNAL(triggered()),this,SLOT(on\_action\_Rank\_triggered()));\*/

//游戏界面初始化

hp = (scores\*)malloc(sizeof(scores));

hq = (scores\*)malloc(sizeof(scores));

/\*t = (scores\*)malloc(sizeof(scores));\*/

for(int i=0;i<4;i++)

{

for(int j=0;j<4;j++)

{

hq->arg[i][j] = 0;

}

}

//随机生成两个数字

int i,j,k, m;

i=0;

j=0;

k=0;

m=0;

while(i==k && j==m)

{

i = rand()%4; /\*x\*/

j = rand()%4; /\*y\*/

k = rand()%4; /\*x\*/

m = rand()%4; /\*y\*/

}

if(hq->arg[4][4] == 0)

{

}

while(hq->arg[i][j]==0 || hq->arg[k][m]==0)

{

hq->arg[i][j] = (rand()%3)\*2;

hq->arg[k][m] = (rand()%3)\*2;

}

hq->next = NULL;

hq->score = 0;

hq->prior = NULL;

playr->head\_score = hq;

//数字显示

show\_shuzi(hq);

show\_score(hq);

show\_best(playr);

for(i=0;i<4;i++)

{

for(j=0; j<4; j++)

{

hp->arg[i][j] = hq->arg[i][j];

}

}

hp->score = hq->score;

hp->next = NULL;

/\*hp->name = q->name;\*/

hp->prior = hq;

hq->next = hp;

hq = hp;

/\*t = hq;\*/

}

MainWindow::~*MainWindow*()

{

delete ui;

}

//重新开始游戏 刷新游戏界面

void MainWindow::on\_action\_New\_triggered()

{

this->deleteLater();

MainWindow\* mainwindow=new MainWindow(NULL);

/\*mainwindow->setGeometry(9,9,600,500);\*/

mainwindow->show();

/\*清空个人该局链表\*/

scores \*hhp,\*hhq;

hhp = playr->head\_score;

playr->head\_score =NULL;

while(hhp->next!=NULL)

{

hhq = hhp->next;

free(hhp);

hhp = hhq;

}

free(hhp);

}

//结束游戏 跳转至over界面

void MainWindow::on\_action\_End\_triggered()

{

Over\* over = new Over(NULL);

over->show();

/\*清空个人该局链表\*/

scores \*hhp, \*hhq;

hhp = playr->head\_score;

playr->head\_score = NULL;

while(hhp != NULL)

{

hhq = hhp->next;//hhq指向hhp指向的结点的下一个结点

free(hhp);

hhp = hhq;

}

free(hhp);

this->close();

}

//弹出rule界面

void MainWindow::on\_action\_Rules\_triggered()

{

Rule\* rule = new Rule(NULL);

rule->show();

}

//弹出rank界面

void MainWindow::on\_action\_Rank\_triggered()

{

Rank\* rank = new Rank(NULL);

rank->show();

}

//弹出best界面

void MainWindow::on\_action\_Best\_triggered()

{

Best\* best = new Best(NULL);

best->show();

}

//向上移动

void MainWindow::move\_up(scores \*p)

{

int i,j;

for(j=0; j<4; j++)

{

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

{

if(p->arg[i][j]!=0 && p->arg[i+1][j]!=0)

{ }

else if(p->arg[i][j]!=0)

{ }

else if(p->arg[i+1][j]!=0)

{

p->arg[i][j] = p->arg[i+1][j];

p->arg[i+1][j] = 0;

}

else

{

if(i+3 == 3)

{

if(p->arg[i+2][j]!=0 && p->arg[i+3][j]!=0)

{

p->arg[i][j] = p->arg[i+2][j];

p->arg[i+1][j] = p->arg[i+3][j];

p->arg[i+2][j] = 0;

p->arg[i+3][j] = 0;

}

else if(p->arg[i+2][j]!=0)

{

p->arg[i][j] = p->arg[i+2][j];

p->arg[i+2][j] = 0;

}

else if(p->arg[i+3][j]!=0)

{

p->arg[i][j] = p->arg[i+3][j];

p->arg[i+3][j] = 0;

}

else

{

}

}

}

}

}

}

//向下移动

void MainWindow::move\_down(scores \*p)

{

int i,j;

for(j=0; j<4; j++)

{

for(i=3; i>0; i--)

{

if(p->arg[i][j]!=0 && p->arg[i-1][j]!=0)

{ }

else if(p->arg[i][j]!=0)

{ }

else if(p->arg[i-1][j]!=0)

{

p->arg[i][j] = p->arg[i-1][j];

p->arg[i-1][j] = 0;

}

else

{

if(i-3 == 0)

{

if(p->arg[i-2][j]!=0 && p->arg[i-3][j]!=0)

{

p->arg[i][j] = p->arg[i-2][j];

p->arg[i-1][j] = p->arg[i-3][j];

p->arg[i-2][j] = 0;

p->arg[i-3][j] = 0;

}

else if(p->arg[i-2][j]!=0)

{

p->arg[i][j] = p->arg[i-2][j];

p->arg[i-2][j] = 0;

}

else if(p->arg[i-3][j]!=0)

{

p->arg[i][j] = p->arg[i-3][j];

p->arg[i-3][j] = 0;

}

else

{

}

}

}

}

}

}

//向左移动

void MainWindow::move\_left(scores \*p)

{

int i,j;

for(i=0; i<4; i++)

{

for(j=0; j<3; j++)

{

if(p->arg[i][j]!=0 && p->arg[i][j+1]!=0)

{ }

else if(p->arg[i][j]!=0)

{ }

else if(p->arg[i][j+1]!=0)

{

p->arg[i][j] = p->arg[i][j+1];

p->arg[i][j+1] =0;

}

else

{

if(j+3 == 3)

{

if(p->arg[i][j+2]!=0 && p->arg[i][j+3]!=0)

{

p->arg[i][j] = p->arg[i][j+2];

p->arg[i][j+1] = p->arg[i][j+3];

p->arg[i][j+2] = 0;

p->arg[i][j+3] = 0;

}

else if(p->arg[i][j+2]!=0)

{

p->arg[i][j] = p->arg[i][j+2];

p->arg[i][j+2] = 0;

}

else if(p->arg[i][j+3]!=0)

{

p->arg[i][j] = p->arg[i][j+3];

p->arg[i][j+3] = 0;

}

}

}

}

}

}

//向右移动

void MainWindow::move\_right(scores \*p)

{

int i,j;

for(i=0; i<4; i++)

{

for(j=3; j>0; j--)

{

if(p->arg[i][j]!=0 && p->arg[i][j-1]!=0)

{ }

else if(p->arg[i][j]!=0)

{ }

else if(p->arg[i][j-1]!=0)

{

p->arg[i][j] =hp->arg[i][j-1];

p->arg[i][j-1] =0;

}

else if(p->arg[i][j-1]!=0)

{

if(j-3 == 0)

{

if(p->arg[i][j-2]!=0 && p->arg[i][j-3]!=0)

{

p->arg[i][j] = p->arg[i][j-2];

p->arg[i][j-1] = p->arg[i][j-3];

p->arg[i][j-2] = 0;

p->arg[i][j-3] = 0;

}

else if(p->arg[i][j-2]!=0)

{

p->arg[i][j] = p->arg[i][j-2];

p->arg[i][j-2] = 0;

}

else if(p->arg[i][j-3]!=0)

{

p->arg[i][j] = p->arg[i][j-3];

p->arg[i][j-3] = 0;

}

}

}

}

}

}

//向上移动后累加

void MainWindow::add\_up(scores \*p)

{

int i,j;

for(j=0; j<4; j++)

{

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

{

if(p->arg[i][j] == p->arg[i+1][j])

{

p->arg[i][j] = p->arg[i][j]+p->arg[i+1][j];

p->score += p->arg[i][j];

p->arg[i+1][j] = 0;

}

else

{ }

}

}

}

//向下移动后累加

void MainWindow::add\_down(scores \*p)

{

int i,j;

for(j=0; j<4; j++)

{

for(i=3; i>0; i--)

{

if(p->arg[i][j] == p->arg[i-1][j])

{

p->arg[i][j] = p->arg[i][j]+p->arg[i-1][j];

p->score += p->arg[i][j];

p->arg[i-1][j] = 0;

}

else

{ }

}

}

}

//向右移动后累加

void MainWindow::add\_right(scores \*p)

{

int i,j;

for(i=0; i<4; i++)

{

for(j=3; j>0; j--)

{

if(p->arg[i][j] == p->arg[i][j-1])

{

p->arg[i][j] = p->arg[i][j]+p->arg[i][j-1];

p->arg[i][j-1] = 0;

p->score += p->arg[i][j];

}

}

}

}

//向左移动后累加

void MainWindow::add\_left(scores \*p)

{

int i,j;

for(i=0; i<4; i++)

{

for(j=0; j<3; j++)

{

if(p->arg[i][j] == p->arg[i][j+1])

{

p->arg[i][j] = p->arg[i][j]+p->arg[i][j+1];

p->arg[i][j+1] = 0;

p->score += p->arg[i][j];

}

else

{ }

}

}

}

//实时显示成绩

void MainWindow::show\_score(scores \*p)

{

ui->SCORE\_1->setText(QString("%1").arg(p->score));

}

//实时显示最好成绩

void MainWindow::show\_best(players \*pp)

{

ui->BEST\_1->setText(QString("%1").arg(pp->best));

}

//实时显示数字

void MainWindow::show\_shuzi(scores \*p)

{

if(p->arg[0][0]!=0)

{

ui->textEdit\_01->setText(QString("%1").arg(p->arg[0][0]));

ui->textEdit\_01->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_01->setText("");

}

if(p->arg[1][0]!=0)

{

ui->textEdit\_02->setText(QString("%1").arg(p->arg[1][0]));

ui->textEdit\_02->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_02->setText("");

}

if(p->arg[2][0]!=0)

{

ui->textEdit\_03->setText(QString("%1").arg(p->arg[2][0]));

ui->textEdit\_03->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_03->setText("");

}

if(p->arg[3][0]!=0)

{

ui->textEdit\_04->setText(QString("%1").arg(p->arg[3][0]));

ui->textEdit\_04->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_04->setText("");

}

if(p->arg[0][1]!=0)

{

ui->textEdit\_05->setText(QString("%1").arg(p->arg[0][1]));

ui->textEdit\_05->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_05->setText("");

}

if(p->arg[1][1]!=0)

{

ui->textEdit\_06->setText(QString("%1").arg(p->arg[1][1]));

ui->textEdit\_06->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_06->setText("");

}

if(p->arg[2][1]!=0)

{

ui->textEdit\_07->setText(QString("%1").arg(p->arg[2][1]));

ui->textEdit\_07->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_07->setText("");

}

if(p->arg[3][1]!=0)

{

ui->textEdit\_08->setText(QString("%1").arg(p->arg[3][1]));

ui->textEdit\_08->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_08->setText("");

}

if(p->arg[0][2]!=0)

{

ui->textEdit\_09->setText(QString("%1").arg(p->arg[0][2]));

ui->textEdit\_09->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_09->setText("");

}

if(p->arg[1][2]!=0)

{

ui->textEdit\_10->setText(QString("%1").arg(p->arg[1][2]));

ui->textEdit\_10->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_10->setText("");

}

if(p->arg[2][2]!=0)

{

ui->textEdit\_11->setText(QString("%1").arg(p->arg[2][2]));

ui->textEdit\_11->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_11->setText("");

}

if(p->arg[3][2]!=0)

{

ui->textEdit\_12->setText(QString("%1").arg(p->arg[3][2]));

ui->textEdit\_12->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_12->setText("");

}

if(p->arg[0][3]!=0)

{

ui->textEdit\_13->setText(QString("%1").arg(p->arg[0][3]));

ui->textEdit\_13->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_13->setText("");

}

if(p->arg[1][3]!=0)

{

ui->textEdit\_14->setText(QString("%1").arg(p->arg[1][3]));

ui->textEdit\_14->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_14->setText("");

}

if(p->arg[2][3]!=0)

{

ui->textEdit\_15->setText(QString("%1").arg(p->arg[2][3]));

ui->textEdit\_15->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_15->setText("");

}

if(p->arg[3][3]!=0)

{

ui->textEdit\_16->setText(QString("%1").arg(p->arg[3][3]));

ui->textEdit\_16->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_16->setText("");

}

}

//游戏成功后最好成绩更新

void MainWindow::best\_update(players \*pp, scores \*p)

{

if(p->score > pp->best)

{

pp->best = p->score;

k = 1;

}

else{

k = 0;

}

}

//游戏成功后重新排名

void MainWindow::rank\_new()

{

//先判断最好成绩是否更新

if(k)

{

players \*h,\*m;

m = (players\*)malloc(sizeof(players));

h = (players\*)malloc(sizeof(players));

//累加计算排名

for(m = head\_1->next; m->next != NULL; )

{

for(h = head\_1->next; h->next != NULL;)

{

m->rank = 1;

if(h->best > m->best)

{

m->rank ++;

}

else{

}

}

}

}

else

{

}

}

//创建新的结点

void MainWindow::create\_pos(scores \*hhq)

{

scores \*hhp;

/\*hhq = hq;\*/

hhp = (scores\*)malloc(sizeof(scores));

for(int i=0;i<4;i++)

{

for(int j=0; j<4; j++)

{

hhp->arg[i][j] = hhq->arg[i][j];

}

}

hhp->score = hp->score;

hhp->next = NULL;

hhp->prior = hhq;

/\*hp->name = q->name;\*/

hp = hhp;

}

//游戏是否结束的判断

void MainWindow::Gameover(scores \*p, players \*q)

{

int i,j,k,m,n,a;

m = 0;

n = 0;

a = 0;

k = 0;

for(i=0; i<4; i++)

{

for(j=0; j<4; j++)

{

if(p->arg[i][j] == 2048)

{

best\_update(q, p);

rank\_new();

//写入文件

QFile file("info.txt");

if(!file.*open*(QIODevice::WriteOnly|QIODevice::Text))

/\*{\*/

qDebug()<</\*"文件打开失败"<<\*/file.errorString();

/\*return ;\*/

/\*}\*/

QTextStream cin(&file);

writeInfo(cin,head\_1);

file.*close*();

//弹出win界面，关闭游戏界面

Win \*win = new Win(NULL);

win->show();

this->close();

a = 1;

}

else if(p->arg[i][j]!=0)

{

k++;

}

}

}

if(k == 16)

{

for(i=0; i<4; i++)

{

for(j=0; j<3; j++)

{

if(p->arg[i][j]!= p->arg[i][j+1])

{

m++;

}

else

{

}

}

}

}

if(m == 9)

{

for(j=0; j<4; i++)

{

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

{

if(p->arg[i][j]!= p->arg[i+1][j])

{

n++;

}

else

{

}

}

}

}

if(n == 9)

{

Over\* over = new Over(0);

over->show();

this->close();

}

else if(a==0)

{

create\_pos(hq);

//随机生成一个数字

i = rand()%4;

j = rand()%4;

while(hp->arg[i][j] != 0)

{

i = rand()%4;

j = rand()%4;

}

for(hp->arg[i][j] = (rand()%3)\*2; hp->arg[i][j] == 0; )

{

hp->arg[i][j] = (rand()%3)\*2;

}

hq = hp;

/\*hp->prior = t;

t->next = hp;\*/

}

else {

}

}

void MainWindow::writeInfo(QTextStream &cin,players \*hhead)

{

players\* p = hhead->next;

while(p){

QString str = p->name +" " + QString::number(p->rank,10)

+ " " +QString::number(p->best,10);// +"\r\n";

cin<<str<<endl;

p = p->next;

}

}

void MainWindow::on\_up\_clicked()

{

/\*t = hp;\*/

move\_up(hp);

move\_up(hp);

add\_up(hp);

show\_score(hp);

move\_up(hp);

show\_shuzi(hp);

Gameover(hp,playr);

show\_shuzi(hp);

}

void MainWindow::on\_left\_clicked()

{

/\*t = hp;\*/

move\_left(hp);

move\_left(hp);

add\_left(hp);

show\_score(hp);

move\_left(hp);

show\_shuzi(hp);

Gameover(hp,playr);

show\_shuzi(hp);

}

void MainWindow::on\_down\_clicked()

{

/\*t = hp;\*/

move\_down(hp);

move\_down(hp);

add\_down(hp);

show\_score(hp);

move\_down(hp);

show\_shuzi(hp);

Gameover(hp,playr);

show\_shuzi(hp);

}

void MainWindow::on\_right\_clicked()

{

/\*t = hp;\*/

move\_right(hp);

move\_right(hp);

add\_right(hp);

show\_score(hp);

move\_right(hp);

show\_shuzi(hp);

Gameover(hp,playr);

show\_shuzi(hp);

}

over.cpp

#include "over.h"

#include "ui\_over.h"

#include "mainwindow.h"

Over::Over(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Over)

{

ui->setupUi(this);

}

Over::~*Over*()

{

delete ui;

}

//重新开始游戏

void Over::on\_pushButton\_clicked()

{

MainWindow \*mainwindow = new MainWindow();

mainwindow->show();

this->deleteLater();

}

//退出游戏

void Over::on\_pushButton\_2\_clicked()

{

this->close();

}

rank.cpp

#include "rank.h"

#include "ui\_rank.h"

extern players \*playr;

Rank::Rank(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Rank)

{

ui->setupUi(this);

ui->label\_3->setText(playr->name);

ui->label\_4->setText(QString("%1").arg(playr->rank));

}

Rank::~*Rank*()

{

delete ui;

}

void Rank::on\_pushButton\_clicked()

{

this->close();

}

rule.cpp

#include "rule.h"

#include "ui\_rule.h"

Rule::Rule(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Rule)

{

ui->setupUi(this);

}

Rule::~*Rule*()

{

delete ui;

}

void Rule::on\_pushButton\_clicked()

{

this->close();

}

shouye.cpp

#include "shouye.h"

#include "ui\_shouye.h"

#include "zhuce.h"

#include "denglu.h"

Shouye::Shouye(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Shouye)

{

ui->setupUi(this);

}

Shouye::~*Shouye*()

{

delete ui;

}

void Shouye::on\_pushButton\_2\_clicked()

{

Zhuce \*zhuce = new Zhuce();

zhuce->show();

this->close();

}

void Shouye::on\_pushButton\_clicked()

{

Denglu \*denglu = new Denglu(NULL);

denglu->show();

this->close();

}

tip.cpp

#include "tip.h"

#include "ui\_tip.h"

Tip::Tip(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Tip)

{

ui->setupUi(this);

}

Tip::~*Tip*()

{

delete ui;

}

void Tip::on\_pushButton\_clicked()

{

this->close();

}

tips.cpp

#include "tips.h"

#include "ui\_tips.h"

Tips::Tips(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Tips)

{

ui->setupUi(this);

}

Tips::~*Tips*()

{

delete ui;

}

void Tips::on\_pushButton\_clicked()

{

this->close();

}

win.cpp

#include "win.h"

#include "ui\_win.h"

#include "denglu.h"

Win::Win(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Win)

{

ui->setupUi(this);

}

Win::~*Win*()

{

delete ui;

}

void Win::on\_pushButton\_clicked()

{

Denglu\* denglu = new Denglu(NULL);

denglu->show();

this->close();

}

void Win::on\_pushButton\_2\_clicked()

{

this->close();

}

zhuce.cpp

#include "zhuce.h"

#include "ui\_zhuce.h"

#include "denglu.h"

#include "shouye.h"

#include "tips.h"

players \*head;//头指针 注册界面

Zhuce::Zhuce(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Zhuce)

{

ui->setupUi(this);

}

Zhuce::~*Zhuce*()

{

delete ui;

}

void Zhuce::on\_pushButton\_clicked()

{

//读入文件

int k = 0;

/\*head = (players\*)malloc(sizeof(players));\*/

head = new players;

head->rank = 0;

/\*head->head\_score = NULL;\*/

head->name = "000";

head->best = 0;

head->next = NULL;

QFile file("G:/2048/GameDesign/info.txt");

if(!file.*open*(QIODevice::ReadOnly|QIODevice::Text))

return;

QTextStream in(&file);

while(!in.atEnd()){

QString line = in.readLine();

readInfo(line);//对读取的一行字符串进行处理

}

//获取信息

QString strText =ui->lineEdit->text();

//比较用户信息

players \*p;

p = new players;

/\*q = new players;\*/

players \*q;

q = head;

p->name = strText;

p->best = 0;

p->rank = 0;

p->next = NULL;

p->head\_score = NULL;

/\*q = head;\*/

for(;q->next != NULL;)

{

if(QString::compare(q->name,p->name))//相等返回0

{

q = q->next;

k = 0;

}

else

{ //存在该用户

//弹出提示界面

Tips \*tips = new Tips();

tips->show();

k = 1;//用户名不符合要求

}

}

//最后一个节点

if(q->next == NULL && QString::compare(q->name,p->name))

{

q->next = p;

k = 0;

}

else

{

//存在该用户

//弹出提示界面

Tips \*tips = new Tips();

tips->show();

k = 1;

}

while(!k)

//写入文件

{

QFile file("info.txt");

if(!file.*open*(QIODevice::WriteOnly|QIODevice::Text))

/\*{\*/

qDebug()<</\*"文件打开失败"<<\*/file.errorString();

/\*return ;\*/

/\*}\*/

QTextStream cin(&file);

writeInfo(cin,head);

file.*close*();

//跳转到登录界面

Denglu \*denglu = new Denglu(NULL);

denglu->show();

this->close();

k = 1;

}

}

void Zhuce::on\_pushButton\_2\_clicked()

{

this->close();

}

void Zhuce::readInfo(QString line)

{

players \*p;

p = new players;

bool ok;

QStringList line\_1;

line\_1 = line.split(" ");

p->name = line\_1.at(0);

p->rank = line\_1.at(1).toInt(&ok,10);

p->best = line\_1.at(2).toInt(&ok,10);

p->next = head->next;

head->next = p;

}

void Zhuce::writeInfo(QTextStream &cin,players \*hhead)

{

players\* p = hhead->next;

while(p){

QString str = p->name +" " + QString::number(p->rank,10)

+ " " +QString::number(p->best,10);// +"\r\n";

cin<<str<<endl;

p = p->next;

}

}

void Zhuce::on\_pushButton\_3\_clicked()

{

Shouye \*shouye = new Shouye;

shouye->show();

this->close();

}

1. 提高部分源程序

back.h

#ifndef BACK\_H

#define BACK\_H

#include <QDialog>

namespace Ui {

class Back;

}

class Back : public QDialog

{

Q\_OBJECT

public:

explicit Back(QWidget \*parent = 0);

~*Back*();

private slots:

void on\_pushButton\_clicked();

private:

Ui::Back \*ui;

};

#endif // BACK\_H

back.cpp

#include "back.h"

#include "ui\_back.h"

Back::Back(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Back)

{

ui->setupUi(this);

}

Back::~*Back*()

{

delete ui;

}

void Back::on\_pushButton\_clicked()

{

this->close();

}

mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "over.h"

#include "rule.h"

#include "win.h"

#include "best.h"

#include "rank.h"

#include "back.h"

#include <time.h>

scores \*hp, \*hq;

scores \*t;

extern players \*playr, \*head\_1;

int k;

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

srand(time(NULL));

ui->setupUi(this);

//状态栏显示 临时信息和永久信息

ui->statusBar->showMessage(tr("新一局游戏现在开始！"));

QLabel \* permanent = new QLabel(this);

permanent->setFrameStyle(QFrame::Box|QFrame::Sunken);

permanent->setText("copyright@turbo");

ui->statusBar->addPermanentWidget(permanent);

//游戏界面初始化

hp = (scores\*)malloc(sizeof(scores));

hq = (scores\*)malloc(sizeof(scores));

/\*t = (scores\*)malloc(sizeof(scores));\*/

for(int i=0;i<4;i++)

{

for(int j=0;j<4;j++)

{

hq->arg[i][j] = 0;

}

}

//随机生成两个数字

int i,j,k, m;

i=0;

j=0;

k=0;

m=0;

while(i==k && j==m)

{

i = rand()%4; /\*x\*/

j = rand()%4; /\*y\*/

k = rand()%4; /\*x\*/

m = rand()%4; /\*y\*/

}

if(hq->arg[4][4] == 0)

{

}

while(hq->arg[i][j]==0 || hq->arg[k][m]==0)

{

hq->arg[i][j] = (rand()%3)\*2;

hq->arg[k][m] = (rand()%3)\*2;

}

hq->next = NULL;

hq->score = 0;

hq->prior = NULL;

playr->head\_score = hq;

//数字显示

show\_shuzi(hq);

show\_score(hq);

show\_best(playr);

for(i=0;i<4;i++)

{

for(j=0; j<4; j++)

{

hp->arg[i][j] = hq->arg[i][j];

}

}

hp->score = hq->score;

hp->next = NULL;

/\*hp->name = q->name;\*/

hp->prior = hq;

hq->next = hp;

/\*hq = hp;\*/

/\*t = hq;\*/

}

MainWindow::~*MainWindow*()

{

delete ui;

}

//重新开始游戏 刷新游戏界面

void MainWindow::on\_action\_New\_triggered()

{

this->deleteLater();

MainWindow\* mainwindow=new MainWindow(NULL);

/\*mainwindow->setGeometry(9,9,600,500);\*/

mainwindow->show();

/\*清空个人该局链表\*/

scores \*hhp,\*hhq;

hhp = playr->head\_score;

playr->head\_score =NULL;

while(hhp->next!=NULL)

{

hhq = hhp->next;

free(hhp);

hhp = hhq;

}

free(hhp);

}

//结束游戏 跳转至over界面

void MainWindow::on\_action\_End\_triggered()

{

Over\* over = new Over(NULL);

over->show();

/\*清空个人该局链表\*/

scores \*hhp, \*hhq;

hhp = playr->head\_score;

playr->head\_score = NULL;

while(hhp != NULL)

{

hhq = hhp->next;//hhq指向hhp指向的结点的下一个结点

free(hhp);

hhp = hhq;

}

free(hhp);

this->close();

}

//弹出rule界面

void MainWindow::on\_action\_Rules\_triggered()

{

Rule\* rule = new Rule(NULL);

rule->show();

}

//弹出rank界面

void MainWindow::on\_action\_Rank\_triggered()

{

Rank\* rank = new Rank(NULL);

rank->show();

}

//弹出best界面

void MainWindow::on\_action\_Best\_triggered()

{

Best\* best = new Best(NULL);

best->show();

}

//悔棋

void MainWindow::on\_action\_Back\_triggered()

{

if(hp->prior != NULL)

{

hp = hp->prior;

hp->next = NULL;

if(hp->prior == NULL)

{

Back \*back = new Back;

back->show();

scores \*hhp;

/\*hhq = hq;\*/

hhp = (scores\*)malloc(sizeof(scores));

for(int i=0;i<4;i++)

{

for(int j=0; j<4; j++)

{

hhp->arg[i][j] = hp->arg[i][j];

}

}

hhp->score = hp->score;

hhp->next = NULL;

hhp->prior = hp;

/\*hp->name = q->name;\*/

hp = hhp;

}

show\_shuzi(hp);

show\_score(hp);

}

else

{

Back \*back = new Back;

back->show();

scores \*hhp;

/\*hhq = hq;\*/

hhp = (scores\*)malloc(sizeof(scores));

for(int i=0;i<4;i++)

{

for(int j=0; j<4; j++)

{

hhp->arg[i][j] = hp->arg[i][j];

}

}

hhp->score = hp->score;

hhp->next = NULL;

hhp->prior = hp;

/\*hp->name = q->name;\*/

hp = hhp;

}

}

//向上移动

void MainWindow::move\_up(scores \*p)

{

int i,j;

for(j=0; j<4; j++)

{

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

{

if(p->arg[i][j]!=0 && p->arg[i+1][j]!=0)

{ }

else if(p->arg[i][j]!=0)

{ }

else if(p->arg[i+1][j]!=0)

{

p->arg[i][j] = p->arg[i+1][j];

p->arg[i+1][j] = 0;

}

else

{

if(i+3 == 3)

{

if(p->arg[i+2][j]!=0 && p->arg[i+3][j]!=0)

{

p->arg[i][j] = p->arg[i+2][j];

p->arg[i+1][j] = p->arg[i+3][j];

p->arg[i+2][j] = 0;

p->arg[i+3][j] = 0;

}

else if(p->arg[i+2][j]!=0)

{

p->arg[i][j] = p->arg[i+2][j];

p->arg[i+2][j] = 0;

}

else if(p->arg[i+3][j]!=0)

{

p->arg[i][j] = p->arg[i+3][j];

p->arg[i+3][j] = 0;

}

else

{

}

}

}

}

}

}

//向下移动

void MainWindow::move\_down(scores \*p)

{

int i,j;

for(j=0; j<4; j++)

{

for(i=3; i>0; i--)

{

if(p->arg[i][j]!=0 && p->arg[i-1][j]!=0)

{ }

else if(p->arg[i][j]!=0)

{ }

else if(p->arg[i-1][j]!=0)

{

p->arg[i][j] = p->arg[i-1][j];

p->arg[i-1][j] = 0;

}

else

{

if(i-3 == 0)

{

if(p->arg[i-2][j]!=0 && p->arg[i-3][j]!=0)

{

p->arg[i][j] = p->arg[i-2][j];

p->arg[i-1][j] = p->arg[i-3][j];

p->arg[i-2][j] = 0;

p->arg[i-3][j] = 0;

}

else if(p->arg[i-2][j]!=0)

{

p->arg[i][j] = p->arg[i-2][j];

p->arg[i-2][j] = 0;

}

else if(p->arg[i-3][j]!=0)

{

p->arg[i][j] = p->arg[i-3][j];

p->arg[i-3][j] = 0;

}

else

{

}

}

}

}

}

}

//向左移动

void MainWindow::move\_left(scores \*p)

{

int i,j;

for(i=0; i<4; i++)

{

for(j=0; j<3; j++)

{

if(p->arg[i][j]!=0 && p->arg[i][j+1]!=0)

{ }

else if(p->arg[i][j]!=0)

{ }

else if(p->arg[i][j+1]!=0)

{

p->arg[i][j] = p->arg[i][j+1];

p->arg[i][j+1] =0;

}

else

{

if(j+3 == 3)

{

if(p->arg[i][j+2]!=0 && p->arg[i][j+3]!=0)

{

p->arg[i][j] = p->arg[i][j+2];

p->arg[i][j+1] = p->arg[i][j+3];

p->arg[i][j+2] = 0;

p->arg[i][j+3] = 0;

}

else if(p->arg[i][j+2]!=0)

{

p->arg[i][j] = p->arg[i][j+2];

p->arg[i][j+2] = 0;

}

else if(p->arg[i][j+3]!=0)

{

p->arg[i][j] = p->arg[i][j+3];

p->arg[i][j+3] = 0;

}

}

}

}

}

}

//向右移动

void MainWindow::move\_right(scores \*p)

{

int i,j;

for(i=0; i<4; i++)

{

for(j=3; j>0; j--)

{

if(p->arg[i][j]!=0 && p->arg[i][j-1]!=0)

{ }

else if(p->arg[i][j]!=0)

{ }

else if(p->arg[i][j-1]!=0)

{

p->arg[i][j] =hp->arg[i][j-1];

p->arg[i][j-1] =0;

}

else if(p->arg[i][j-1]!=0)

{

if(j-3 == 0)

{

if(p->arg[i][j-2]!=0 && p->arg[i][j-3]!=0)

{

p->arg[i][j] = p->arg[i][j-2];

p->arg[i][j-1] = p->arg[i][j-3];

p->arg[i][j-2] = 0;

p->arg[i][j-3] = 0;

}

else if(p->arg[i][j-2]!=0)

{

p->arg[i][j] = p->arg[i][j-2];

p->arg[i][j-2] = 0;

}

else if(p->arg[i][j-3]!=0)

{

p->arg[i][j] = p->arg[i][j-3];

p->arg[i][j-3] = 0;

}

}

}

}

}

}

//向上移动后累加

void MainWindow::add\_up(scores \*p)

{

int i,j;

for(j=0; j<4; j++)

{

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

{

if(p->arg[i][j] == p->arg[i+1][j])

{

p->arg[i][j] = p->arg[i][j]+p->arg[i+1][j];

p->score += p->arg[i][j];

p->arg[i+1][j] = 0;

}

else

{ }

}

}

}

//向下移动后累加

void MainWindow::add\_down(scores \*p)

{

int i,j;

for(j=0; j<4; j++)

{

for(i=3; i>0; i--)

{

if(p->arg[i][j] == p->arg[i-1][j])

{

p->arg[i][j] = p->arg[i][j]+p->arg[i-1][j];

p->score += p->arg[i][j];

p->arg[i-1][j] = 0;

}

else

{ }

}

}

}

//向右移动后累加

void MainWindow::add\_right(scores \*p)

{

int i,j;

for(i=0; i<4; i++)

{

for(j=3; j>0; j--)

{

if(p->arg[i][j] == p->arg[i][j-1])

{

p->arg[i][j] = p->arg[i][j]+p->arg[i][j-1];

p->arg[i][j-1] = 0;

p->score += p->arg[i][j];

}

}

}

}

//向左移动后累加

void MainWindow::add\_left(scores \*p)

{

int i,j;

for(i=0; i<4; i++)

{

for(j=0; j<3; j++)

{

if(p->arg[i][j] == p->arg[i][j+1])

{

p->arg[i][j] = p->arg[i][j]+p->arg[i][j+1];

p->arg[i][j+1] = 0;

p->score += p->arg[i][j];

}

else

{ }

}

}

}

//实时显示成绩

void MainWindow::show\_score(scores \*p)

{

ui->SCORE\_1->setText(QString("%1").arg(p->score));

}

//实时显示最好成绩

void MainWindow::show\_best(players \*pp)

{

ui->BEST\_1->setText(QString("%1").arg(pp->best));

}

//实时显示数字

void MainWindow::show\_shuzi(scores \*p)

{

if(p->arg[0][0]!=0)

{

ui->textEdit\_01->setText(QString("%1").arg(p->arg[0][0]));

ui->textEdit\_01->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_01->setText("");

}

if(p->arg[1][0]!=0)

{

ui->textEdit\_02->setText(QString("%1").arg(p->arg[1][0]));

ui->textEdit\_02->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_02->setText("");

}

if(p->arg[2][0]!=0)

{

ui->textEdit\_03->setText(QString("%1").arg(p->arg[2][0]));

ui->textEdit\_03->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_03->setText("");

}

if(p->arg[3][0]!=0)

{

ui->textEdit\_04->setText(QString("%1").arg(p->arg[3][0]));

ui->textEdit\_04->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_04->setText("");

}

if(p->arg[0][1]!=0)

{

ui->textEdit\_05->setText(QString("%1").arg(p->arg[0][1]));

ui->textEdit\_05->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_05->setText("");

}

if(p->arg[1][1]!=0)

{

ui->textEdit\_06->setText(QString("%1").arg(p->arg[1][1]));

ui->textEdit\_06->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_06->setText("");

}

if(p->arg[2][1]!=0)

{

ui->textEdit\_07->setText(QString("%1").arg(p->arg[2][1]));

ui->textEdit\_07->setFont(QFont(tr("Consolas"), 50));

}

else {

ui->textEdit\_07->setText("");

}

if(p->arg[3][1]!=0)

{

ui->textEdit\_08->setText(QString("%1").arg(p->arg[3][1]));

ui->textEdit\_08->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_08->setText("");

}

if(p->arg[0][2]!=0)

{

ui->textEdit\_09->setText(QString("%1").arg(p->arg[0][2]));

ui->textEdit\_09->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_09->setText("");

}

if(p->arg[1][2]!=0)

{

ui->textEdit\_10->setText(QString("%1").arg(p->arg[1][2]));

ui->textEdit\_10->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_10->setText("");

}

if(p->arg[2][2]!=0)

{

ui->textEdit\_11->setText(QString("%1").arg(p->arg[2][2]));

ui->textEdit\_11->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_11->setText("");

}

if(p->arg[3][2]!=0)

{

ui->textEdit\_12->setText(QString("%1").arg(p->arg[3][2]));

ui->textEdit\_12->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_12->setText("");

}

if(p->arg[0][3]!=0)

{

ui->textEdit\_13->setText(QString("%1").arg(p->arg[0][3]));

ui->textEdit\_13->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_13->setText("");

}

if(p->arg[1][3]!=0)

{

ui->textEdit\_14->setText(QString("%1").arg(p->arg[1][3]));

ui->textEdit\_14->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_14->setText("");

}

if(p->arg[2][3]!=0)

{

ui->textEdit\_15->setText(QString("%1").arg(p->arg[2][3]));

ui->textEdit\_15->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_15->setText("");

}

if(p->arg[3][3]!=0)

{

ui->textEdit\_16->setText(QString("%1").arg(p->arg[3][3]));

ui->textEdit\_16->setFont(QFont(tr("Consolas"), 50));

}

else{

ui->textEdit\_16->setText("");

}

}

//游戏成功后最好成绩更新

void MainWindow::best\_update(players \*pp, scores \*p)

{

if(p->score > pp->best)

{

pp->best = p->score;

k = 1;

}

else{

k = 0;

}

}

//游戏成功后重新排名

void MainWindow::rank\_new()

{

//先判断最好成绩是否更新

if(k)

{

players \*h,\*m;

m = (players\*)malloc(sizeof(players));

h = (players\*)malloc(sizeof(players));

//累加计算排名

for(m = head\_1->next; m->next != NULL; )

{

for(h = head\_1->next; h->next != NULL;)

{

m->rank = 1;

if(h->best > m->best)

{

m->rank ++;

}

else{

}

}

}

}

else

{

}

}

//创建新的结点

void MainWindow::create\_pos()

{

/\*hq = hp;\*/

scores \*hhp;

/\*hhq = hq;\*/

hhp = (scores\*)malloc(sizeof(scores));

for(int i=0;i<4;i++)

{

for(int j=0; j<4; j++)

{

hhp->arg[i][j] = hp->arg[i][j];

}

}

hhp->score = hp->score;

hhp->next = NULL;

hhp->prior = hp;

/\*hp->name = q->name;\*/

hp = hhp;

}

//游戏是否结束的判断

void MainWindow::Gameover(scores \*p, players \*q)

{

int i,j,k,m,n,a;

m = 0;

n = 0;

a = 0;

k = 0;

for(i=0; i<4; i++)

{

for(j=0; j<4; j++)

{

if(p->arg[i][j] == 2048)

{

best\_update(q, p);

rank\_new();

//写入文件

QFile file("info.txt");

if(!file.*open*(QIODevice::WriteOnly|QIODevice::Text))

/\*{\*/

qDebug()<</\*"文件打开失败"<<\*/file.errorString();

/\*return ;\*/

/\*}\*/

QTextStream cin(&file);

writeInfo(cin,head\_1);

file.*close*();

//弹出win界面，关闭游戏界面

Win \*win = new Win(NULL);

win->show();

this->close();

a = 1;

}

else if(p->arg[i][j]!=0)

{

k++;

}

}

}

if(k == 16)

{

for(i=0; i<4; i++)

{

for(j=0; j<3; j++)

{

if(p->arg[i][j]!= p->arg[i][j+1])

{

m++;

}

else

{

}

}

}

}

if(m == 9)

{

for(j=0; j<4; i++)

{

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

{

if(p->arg[i][j]!= p->arg[i+1][j])

{

n++;

}

else

{

}

}

}

}

if(n == 9)

{

Over\* over = new Over(0);

over->show();

this->close();

}

else if(a==0)

{

/\*create\_pos(hq);\*/

//随机生成一个数字

i = rand()%4;

j = rand()%4;

while(hp->arg[i][j] != 0)

{

i = rand()%4;

j = rand()%4;

}

for(hp->arg[i][j] = (rand()%3)\*2; hp->arg[i][j] == 0; )

{

hp->arg[i][j] = (rand()%3)\*2;

}

create\_pos();

/\*hp->prior = t;

t->next = hp;\*/

}

else {

}

}

void MainWindow::writeInfo(QTextStream &cin,players \*hhead)

{

players\* p = hhead->next;

while(p){

QString str = p->name +" " + QString::number(p->rank,10)

+ " " +QString::number(p->best,10);// +"\r\n";

cin<<str<<endl;

p = p->next;

}

}

void MainWindow::on\_up\_clicked()

{

/\*t = hp;\*/

move\_up(hp);

move\_up(hp);

add\_up(hp);

show\_score(hp);

move\_up(hp);

show\_shuzi(hp);

Gameover(hp,playr);

show\_shuzi(hp);

}

void MainWindow::on\_left\_clicked()

{

/\*t = hp;\*/

move\_left(hp);

move\_left(hp);

add\_left(hp);

show\_score(hp);

move\_left(hp);

show\_shuzi(hp);

Gameover(hp,playr);

show\_shuzi(hp);

}

void MainWindow::on\_down\_clicked()

{

/\*t = hp;\*/

move\_down(hp);

move\_down(hp);

add\_down(hp);

show\_score(hp);

move\_down(hp);

show\_shuzi(hp);

Gameover(hp,playr);

show\_shuzi(hp);

}

void MainWindow::on\_right\_clicked()

{

/\*t = hp;\*/

move\_right(hp);

move\_right(hp);

add\_right(hp);

show\_score(hp);

move\_right(hp);

show\_shuzi(hp);

Gameover(hp,playr);

show\_shuzi(hp);

}