**长沙理工大学**

**《软件系统设计基础实训》报告**

**俄罗斯方块**

张志刚



**学 院 计算机与通信工程 专 业 软件工程**

**班 级 软件13-1 学 号 201316080131**

**学生姓名 张志刚 指导教师 项洁**

**课程成绩 完成日期 2015年7月18日**

**实训成绩评定**

**学 院 计算机与通信工程 专 业 软 件 工 程**

**班 级 软件13-1 学 号 201316080131**

**学生姓名 张志刚 指导教师 项 洁**

**课程成绩 完成日期 2015.7.18**

**指导教师对学生在实训中的评价**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 评分项目 | 优 | 良 | 中 | 及格 | 不及格 |
| 实训中的创造性成果 |  |  |  |  |  |
| 学生掌握课程内容的程度 |  |  |  |  |  |
| 实训完成情况 |  |  |  |  |  |
| 实训动手能力 |  |  |  |  |  |
| 文字表达 |  |  |  |  |  |
| 学习态度 |  |  |  |  |  |
| 规范要求 |  |  |  |  |  |
| 实训论文的质量 |  |  |  |  |  |

**指导教师对实训的评定意见**

|  |
| --- |
| 综合成绩 指导教师签字 年 月 日 |

**实训任务书**

计算机与通信工程学院 软件工程专业

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 课程名称 | 软件系统设计  基础实训 | 时间 | 2015～2016学年第2学期18～20周 | |
| 学生姓名 | 张志刚 | 指导老师 | | 项洁 |
| 题 目 | 俄罗斯方块  ——俄罗斯方块和数据库维护的设计与实现 | | | |
| 主要内容：  本实训任务要求完成俄罗斯方块的设计与实现，本系统分为四个大模块，一人完成，其中本实训任务需要完成以下模块：   1. 角色管理模块。其中角色的创建。 2. 核心游戏模块：本模块用来实现传统游戏的功能 3. 记录管理模块：本模块用来管理玩家的游戏记录 4. 绘图模块：本模块用于实现整个游戏中的绘图 | | | | |
| 要求：  （1）能独立地运用程序设计语言和数据库方面知识，编制一个规模适当的应用软件系统。具体包括对系统的需求分析、设计、编码、测试以及编写各种文档的工作。  （2）按要求编写实训报告书，能正确编写分析、设计、编码、测试等技术文档。  （3）通过实训培养学生严谨的科学态度，认真的工作作风和团队协作精神。  （4）学生应抱着严谨认真的态度积极投入到实训过程中。  （5）在老师的指导下，要求学生完成实训的全部内容。 | | | | |
| 应当提交的文件：  （1）实训日记  （2）实训报告。  （3）实训附件（源程序、各类图纸、实验数据、运行截图等）。 | | | | |

俄罗斯方块

**学生姓名：张志刚 指导老师：项洁**

摘 要

俄罗斯方块游戏是俄罗斯人阿莱克斯帕伊特诺夫在上世纪80年代末开发出来，以其简单的游戏规则，以及变幻无穷的游戏情节，并且能在游戏中益智等特点而成为家喻户晓，老少皆宜的经典游戏。

游戏实现是有4个相同颜色的小正方形的色块组成的7个不同形状的方块随机的从游戏界面顶端下落，并且可以通过键盘的4个方向键来控制游戏的旋转，左右移动，加速移动操作。在游戏过程中，玩家可以用这7种不同形状的方块拼积木，消除积木得分。从软件问题定义开始，接着进行可行性研究分析、需求分析、概要设计、详细设计、最后对程序进行测试，整个开发过程贯穿软件开发的知识体系。这款软件具有丰富的游戏操作功能，用户可以根据自己的操作水平和喜好进行相应的游戏操作设置，从而不仅游戏的初学者也能从中享受乐趣，即便高水平的玩家也能从游戏中尝到挑战的刺激。该游戏在visual studio 2010环境中使用C#以及sql server2008开发实现。C#是基于面向对象的开发设计编程语言，面向对象的开发在近年已广为普及，成为软件开发最基本的方法。

关键词 俄罗斯方块；面向对象；C#；

Tetris

Student name:ZHANG ZHI-GANG Advisor：XIANG JIE

**Abstract**

Tetris game is that the russians alex kuyt's developed in the late 1980 s, with its simple rules of the game, and change the endless game plot, and educational characteristics and ecame a household name in the game, a classic game for young and old.

Game implementation is four shades of the same color of the small square of seven different in the shape of a square random fall from the top of the game interface, and can be through the four direction of the keyboard keys to control the rotation of the game, moving from left to right, accelerating mobile operations.In the process of the game, players can use the seven kinds of different shape of square blocks, eliminate the blocks.From a software problem definition, then analyzed feasibility study, demand analysis, general design, detailed design, finally testing procedures, throughout the entire development process of software development knowledge.This software has the rich game operating functions, users can according to their own operation level and preferences for the corresponding game operation Settings, thus not only the beginners can also can enjoy the pleasure of the game, even if the high level players can also stimulate the taste out of the game the challenge.The game in the visual studio 2010 environment using c # and SQL server2008 development implementation.C # is based on the development design of object-oriented programming language, object-oriented development in recent years has been widely popular, became the most basic software development method.

**Key words** Tetris; [object-oriented](javascript:void(0););C#

目录

[1项目概述 1](#_Toc424725772)

[1.1项目开发的意义 1](#_Toc424725773)

[1.2项目开发的平台 1](#_Toc424725774)

[2项目需求分析 2](#_Toc424725775)

[2.1俄罗斯方块的规则 2](#_Toc424725776)

[2.2功能分析 2](#_Toc424725777)

[2.3数据流图 2](#_Toc424725778)

[3数据库设计 3](#_Toc424725779)

[3.1需求分析 3](#_Toc424725780)

[3.2概念结构设计 4](#_Toc424725781)

[3.3逻辑结构设计 5](#_Toc424725782)

[3.4数据库构建语句 5](#_Toc424725783)

[4项目设计与实现 6](#_Toc424725784)

[4.1 项目的模块划分 6](#_Toc424725785)

[4.2图形的数据结构 7](#_Toc424725786)

[4.3设计的主要原理 8](#_Toc424725787)

[4.4项目的界面展示 9](#_Toc424725788)

[参考文献 13](#_Toc424725789)

[附录 14](#_Toc424725790)

[A 方块旋转源码 14](#_Toc424725791)

[B 处理键盘消息事件 15](#_Toc424725792)

[C 创建角色源码 16](#_Toc424725793)

[D 数据库操作源码 17](#_Toc424725794)

1项目概述

1.1项目开发的意义

该课程设计的目的是实现经典模式的所有功能，并且运用数据库知识，实现玩家创建、记录存储、记录检索、记录删除、保存游戏等功能。玩家可以通过此系统，在任意时刻保存游戏，与此同时，将当前的成绩存储作为一条记录，并且可以以任意一条保存记录继续游戏，重现保存时的游戏状态；玩家也可以在排行榜上查询自己的历史成绩，以及和其他玩家的成绩pk后的排行榜。使玩家得到更好的体验。

1.2项目开发的平台

课程设计的软件平台：

主要开发环境：Vs2010

数据库开发环境：sql server2008

数据库设计：visio2013

课程设计的硬件平台：

联想笔记本电脑

2项目需求分析

为了开发出真正满足用户需求的程序，尤其是游戏程序，首先必须要知道用户的需求，把用户的需求作为前提，来指导程序的开发。

2.1俄罗斯方块的规则

俄罗斯方块是由小方块组成的不同形状的方块陆续从屏幕上方落下来，玩家通过调整方块的位置和形状，使它们在屏幕底部拼出完整的一 条或几条由方块行。这些完整的行会立即消失，给新落下来的板块腾出空间，与此同时，玩家得到分数奖励。没有被消除掉的方块不断堆积起来，在游戏的过程中，随着玩家的分数和等级增高，游戏的难度会随之加大，即方块下落的速度会加快，一旦方块堆到屏幕顶端，玩家便告输，游戏结束。

游戏的得分规则,规定一次消一行得5分，一次消两行得15分，一次消3行得25分，一次消4行得35分，并且，游戏中一次最多消4行，最少消0行。

游戏的难度变化规则，规定游戏分数每达到100的倍数时游戏的级别会增1，并且难度系数加1.

游戏界面规定，一个用于摆放小型正方形的平面虚拟场地，即游戏界面，其标准大小：行宽为14，列高为20，以每个小正方形为单位。并且一组由4个小型正方形组成的规则图形（方块），共有7种如图2-1，分别以可以简单的以S、Z、L、J、I、O、T这7个字母的形状来命名这7种方块。

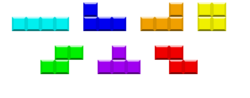
[](http://baike.baidu.com/picture/28513/15345645/0/64380cd7912397dd96df99415982b2b7d1a287d4.html?fr=lemma&ct=single)

图2.1 7种方块图形

2.2功能分析

（1）在游戏过程中没有任何操作时自动下降

（2）在游戏过程中按下方向键能够实现加速下降

（3）在游戏过程中按上方向键能够实现方块旋转

（4）在游戏过程中按左右方向键能够实现方块的左右移动

（5）能够显示游戏当前的分数和级别

（6）在游戏过程中可以播放背景音乐

（7）创建一个玩家，用来记录该玩家的所有游戏记录

（8）玩家可以在任意游戏时刻暂停游戏、保存游戏

（9）玩家可以看自己的历史成绩以及和其它玩家的综合排名

（10）玩家可以删除某个历史成绩

2.3数据流图



图2.2 项目的0层图



图2.3 项目中创建角色的1层图



图2.4 项目中核心游戏、查看成绩、排行榜的1层图



图2.5 项目中保存游戏、继续游戏的1层图

3数据库设计

3.1需求分析

根据数据流图列出部分数据字典如下：

数据流名：注册信息

简述：角色的信息

组成：角色姓名+创建时间

数据流名：游戏记录

简述：游戏的成绩

组成：分数+级数+记录时间

数据流名：游戏信息

简述：游戏的信息

组成：当前方块的所有左标+下一个方块的所有坐标+已固定的方块的位置信息

数据流名：存储信息

简述：保存游戏时要保存的信息

组成：游戏信息+玩家信息+游戏记录

数据流名：玩家信息

简述：玩家的信息

组成：角色的姓名

3.2概念结构设计



图3.1 ER图

3.3逻辑结构设计

根据概念设计阶段的ER图，可以得到如下所示的数据库逻辑结构

表3-2 角色表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实体 | 属性 | 数据类型 | 描述 | 是否为空 |
| 角色 | 角色Id | Smallint | 主键，自动增长 | 非空 |
| 角色名字 | Varchar（12） |  | 非空 |
| 创建时间 | Varchar（20） |  | 非空 |

表3-3记录表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实体 | 属性 | 数据类型 | 描述 | 是否为空 |
| 记录 | 记录Id | Smallint | 主键，自动增长 | 非空 |
| 角色Id | Smallint |  | 非空 |
| 记录时间 | Varchar（20） |  | 非空 |
| 分数 | Int |  | 非空 |
| 级别 | Int |  | 非空 |

表3-4游戏表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实体 | 属性 | 数据类型 | 描述 | 是否为空 |
| 游戏 | 游戏Id | Smallint | 主键，自动增长 | 非空 |
| 当前方块坐标 | Varchar（8） |  | 非空 |
| 下个方块坐标 | Varchar（8） |  | 非空 |
| 以固定方块位置 | Varchar（280） |  | 非空 |
| 保存时间 | Varchar（20） |  | 非空 |

3-5存储表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实体 | 属性 | 数据类型 | 描述 | 是否为空 |
| 存储 | 存储Id | Smallint | 主键，自动增长 | 非空 |
| 记录Id | Smallint |  | 非空 |
| 角色Id | Smallint |  | 非空 |
| 游戏Id | Smallint |  | 非空 |

3.4数据库构建语句

//创建数据库

create database tetrix;

use tetrix;

//创建角色表

create table roles(

roleId smallint not null identity(1,1) primary key,

roleName varchar(12) not null,

createdtime varchar(20) not null

);

//创建记录表

create table records(

recordId smallint not null primary key identity(1,1),

roleId smallint not null,

recordtime varchar(20) not null,

score int not null,

levels int not null

);

//创建游戏表

create table games(

gameId smallint not null identity(1,1) primary key,

curCoords varchar(8) not null,

nextCoords varchar(8) not null,

state varchar(280) not null，

saveTime varchar(20) not null

);

//创建存储表

create table storage(

stoId smallint not null identity(1,1) primary key ,

recordId smallint not null,

roleId smallint not null,

gameId smallint not null

);

4项目设计与实现

在这个游戏中主要分了4个模块：

1. 角色管理模块：本模块用来负责游戏中角色的管理
2. 核心游戏模块：本模块用来实现传统游戏的功能
3. 记录管理模块：本模块用来管理玩家的游戏记录
4. 绘图模块：本模块用于实现整个游戏中的绘图
   1. 项目的模块划分



图4.1 项目的H图

4.2图形的数据结构

该项目的数据结构主要分为两部分，第一部分是方块定义，该项目中一共定义了7个方块，其中每个方块由每个4个小正方形组成，且每个小正方形用其左上角的坐标显示，因此一个方块可以用一个2维数组表示(坐标是相对于mainWindow左上角的)，并且这7种方块和左边的对应关系如下所示：

I型方块：I = { { 0, 1}, { 0, 2 }, { 0, 3 }, { 0, 4}}



图4.2 I方块的形状

S型方块：S={{0,0},{0,1},{1,1},{1,2}}



图4.3 S方块的形状

T型方块：T = {{0,1},{1,1},{1, 0},{1,2}}



图4.4 T方块的形状

Z型方块：Z = {{0,1},{0,2},{1,0},{1,1}}



图4.5 Z方块的形状

J型方块：J = {{0，2},{1,0},{1, 2},{1, 2}}



图4.6 J方块的形状

L型方块：L = {{0,0},{ 0, 1},{0, 2},{ 2, 1}}



图4.7 L方块的形状

O型方块：O = {{0, 0},{1, 0},{0, 1},{1, 1}}



图4.8 O方块的形状

第二部分是背景图，用来存储以固定方块的信息，该项目用了一个20\*14的二维数组表示，且每一个数组元素代表的是一个小正方形。固定方块信息指该位置是否已被下落的方块给占用，该项目用8种颜色来表明该位置是否被占用，以及被那种类型的方块给占用，这里用颜色：YellowGreen、Red、Green、Yello、Purple、Blue、gray、Pink依次代表N、Z、S、L、T、O、J、I方块，另外还有一种颜色即白色，来表示该位置没有别方块占用，并且将颜色定义成枚举变量。

4.3设计的主要原理

1.方块的旋转

在游戏中，当玩家按上方向键时，方块会发生逆时针运算，旋转原理如下：

由于该项目是以方块中的4个小方形左上坐标来标识整个方块的，将方块逆时针旋转相当于将4个小正方形的左上角坐标的每个都绕某个点O逆时针旋转90度，得到的4坐标即是旋转后的方块的4个小正方形坐标，这里取的O点坐标为4个小正方形中的最大x坐标和最小x坐标的平均值，以及最大y坐标和最小y坐标的平均值。设待旋转的为点A(x,y)。此时，是绕点O(p,q)旋转，则由数学知识可以计算出旋转后的坐标为A’(y-q+p, p-x+q)。具体源码参见附录A ；

2.加速下降

在游戏中，当玩家按下方向键时，方块会加速下降，加速下降原理如下：

程序处理下方向键的事件是，程序进入循环，单单调用方块下降函数忽略程序的timer，在循环中如果下到最底端，则程序跳出循环，继续进行程序。源码见附录B

3.方块的左移、右移，

在游戏中，当玩家按左、右方向键时，方块会像左、右移动一个小正方形，移动原理如下：

程序处理左方向键的事件时，将当前方块的每个小方块的横坐标减一，当处理有方向键时，将当前方块的每个小方块的横坐标加一。源码见附录B

4.创建角色

当游戏结束时，会弹出一个对gameover的对话框，可以显示本次记录的分数

级别，以及提示玩家输入自己的名字，当玩家输入自己的名字后点击确定，程序会根据检索数据库来判断是要简单的给玩家增加记录还是给创建新的玩家，与此同时，该对话框会显示该玩家的本次记录在所有记录中排名。源码见附录C

5.保存记录

在游戏过程中，当玩家点击保存游戏时，会弹出一个saveGame的对话框，可以

显示当前的分数以及级数，以及提示玩家输入自己的名字，当玩家输入自己的名字后点击确定，程序会根据检索数据库来判断是要简单的给玩家增加记录还是给创建新的玩家，与此同时，程序会将当前游戏的信息以及记录和角色的信息存储到数据库中。

6.继续游戏

打开游戏时，玩家可以选择继续游戏，这个功能使玩家能够从以前任意一次保存

的游戏开始玩

6.删除记录

在显示当前用户的记录是，用户可以选择删除记录。

4.4项目的界面展示

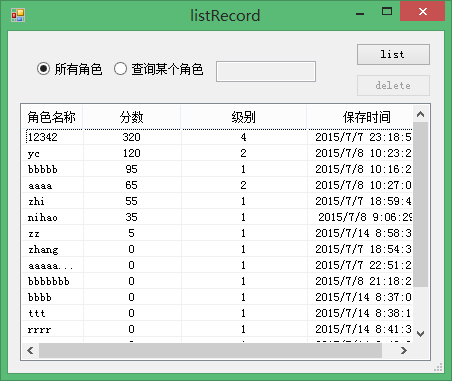


图4.9游戏记录的排行榜

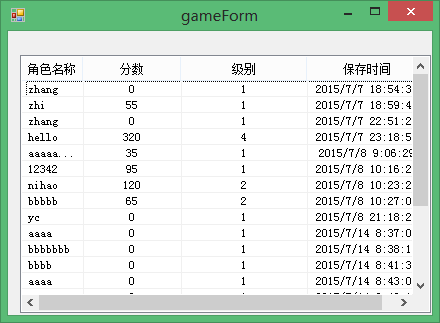


图4.10游戏记录的保存记录

****

图4.11游戏的保存界面

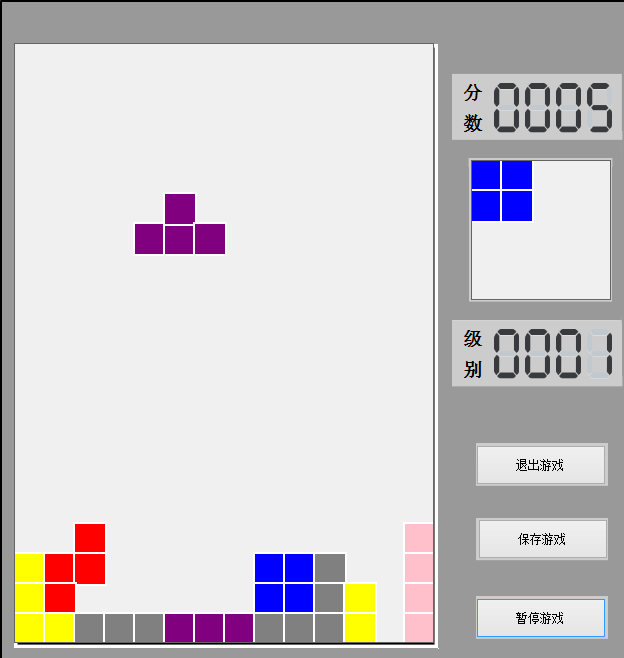
****

图4.12游戏图

参考文献

[1] 刘先峰，曹步闻，李高仕**.**  数据库系统原理与应用[M].第一版第一次**. 华中科技大学出版社**，2012年9月

[2] 齐立波，黄静**.**  C#经典入门[M].第四版第一次**. 清华大学出版社**，2008年12月

[3] 李铭，黄静**.**  C#经典高级编程[M].第六版第一次**. 清华大学出版社**，2008年10月

[4] Michael Milton**.**  译者：李芳 深入浅出数据分析 **清华大学出版社**，2009年

[5] Abraham Silberschatz,Henry F.Korth,S.Sudarshan**.**  译者：杨冬青，李红燕，唐世谓 数据库系统概念[M].第六版第一次**. 机械工业出版社**，2012年3月

附录

A 方块旋转源码

private void changeTetrix(int[,] curTetrix) {

/\*功能：用于改变当前方块的形状，当按上方向键时

\*当前方块顺时针旋转90度

\*算法：根据数学公式，将一点A（x,y）绕原点旋转90度得

\*新的点（y,-x）,当绕的点O(p,q)不是原点时可以先将原点移到O点

\*处，此时A的坐标为(（x-p）,(y-q))，旋转后变成（（y-q）,(p-x)）

\*则原位置应该为(（y-q+p）,(p-x+q))

\*分析的将O点设为方块中四个小方块中左上角最大x最大y坐标对应的坐标（(xMax+xMin)/2，(yMax+yMin)/2）\*/

int xMax = maxX(curTetrix);

int yMax = -maxY(curTetrix);

int xMin = minX(curTetrix);

int yMin = -minY(curTetrix);

int xTemp = (xMax + xMin) / 2 +1;

int yTemp = (yMax + yMin) / 2;

int[,] newTetrix = new int[4,2];

if (xTemp != 1 && yTemp != 0) {

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

{

newTetrix[i, 0] = -curTetrix[i, 1] - yTemp + xTemp;

newTetrix[i, 1] = -(xTemp - curTetrix[i, 0] + yTemp) + 1;

if (newTetrix[i, 0] < 0 || newTetrix[i, 1] < 0)

{

MessageBox.Show("Error!");

}

}

}

if (checkRight(newTetrix) && checkLeft(newTetrix) &&

checkDown(newTetrix) && checkUp(newTetrix))

{

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

curTetrix[i, 0] = newTetrix[i, 0] - 1;

curTetrix[i, 1] = newTetrix[i, 1];

}

}

}

B 处理键盘消息事件

protected override bool ProcessDialogKey(Keys keyData){

/\*功能：处理键盘消息

\*UP：旋转方块

\*Right：向右移动方块

\*Left：向左移动方块

\*Down：加速下降\*/

switch (keyData){

case Keys.Up:

changeTetrix(curTetrix);

drawImage();

break;

case Keys.Left:

if (checkLeft(curTetrix)){

//检查通过,将方块中的4个小方块的各横坐标减1

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

curTetrix[i, 0] -= 1;

}

}

drawImage();

break;

case Keys.Right:

if (checkRight(curTetrix)){

//检查通过,将方块中的4个小方块的各横坐标减1

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

curTetrix[i, 0] += 1;

}

}

drawImage();

break;

case Keys.Down:

for (int i = 20; i >= 0; i--){

if (!downTetrix(curTetrix)) {

break;

}

}

break;

}

if (keyData == Keys.Up || keyData == Keys.Down ||

keyData == Keys.Left || keyData == Keys.Right){

return false;

}

else{

return base.ProcessDialogKey(keyData); }}

C 创建角色源码

private void submit\_Click(object sender, EventArgs e)

{

String name = this.name.Text;

String time = DateTime.Now.ToString();

Int16 roleId;

int score = Convert.ToInt32(this.scoreDisp.Text);

int levels = Convert.ToInt32(this.rankDisp.Text);

if (this.submit.Text == "返回主菜单") {

this.submit.Text = "确定";

pa3.pa2.pa1.Show();

this.Close();

return;

}

String sql = String.Format("select \* from roles where roles.roleName='{0}'", name);

if (String.IsNullOrEmpty(name))

{

MessageBox.Show("姓名不为空！");

return ;

}

reader = readerCommand(sql);

if (reader.Read())

{

roleId = reader.GetInt16(0);

closeConn();

sql = String.Format("insert into records(roleId, recordtime, score, levels) values('{0}', '{1}','{2}','{3}')", roleId, time, score, levels);

queryCommand(sql);

}

else {

closeConn();

sql = String.Format("insert into roles(roleName, createdtime) values('{0}', '{1}')", name, time);

queryCommand(sql);

sql = String.Format("select \* from roles where roles.roleName='{0}'", name);

reader = readerCommand(sql);

if (reader.Read())

{

roleId = reader.GetInt16(0);

closeConn();

sql = String.Format("insert into records(roleId, recordtime, score, levels) values('{0}', '{1}','{2}','{3}')", roleId, time, score, levels);

queryCommand(sql);

}

}

//给排名赋值

int ranking = getRanking(time);

this.rankDisp.Text = ranking.ToString();

this.submit.Text = "返回主菜单";

this.lab.Text = "排名";

this.name.ReadOnly = true;

}

D 数据库操作源码

class db

{

private SqlConnection conn; //用于记录数据库资源

private SqlCommand sqlcom; //用来记录数据库命令对象

private String sql; //用来存储数据库命令字符串

public db() {

conn = getConn();

}

public SqlConnection getConn()

{

/\*功能：用于连接数据库\*/

string connString = String.Format(@"Data Source = localhost;

Initial Catalog=Tetrix;

Integrated Security=True");

SqlConnection conn = new SqlConnection(connString);

return conn;

}

public void queryCommand(string sql)

{

/\*功能：用于执行sql语句以ExcuteNonQury方法

\*针对Update、Insert、Delete方法\*/

SqlConnection conn = this.getConn();

conn.Open();

SqlCommand sqlcom = new SqlCommand(sql, conn);

sqlcom.ExecuteNonQuery();

sqlcom.Dispose();

conn.Close();

conn.Dispose();

}

public SqlDataReader readerCommand(string sql)

{

/\*功能：用于执行sql语句以ExcuteReader方法

\*针对select方法\*/

conn = this.getConn();

conn.Open();

sqlcom = new SqlCommand(sql, conn);

SqlDataReader reader = sqlcom.ExecuteReader();

return reader;

}

public void closeConn()

{

sqlcom.Dispose();

conn.Close();

conn.Dispose();

}

}