南京郵電大學

实 验 报 告

（2017 / 2018 学年 第 一 学期）

课程名称	Windows 高级软件开发
实验名称	俄罗斯方块
实验时间	2017 年 12 月 28 日
指导单位	南京邮电大学
指导教师	张卫丰

学生姓名	邵萍	班级学号	B15041710
学院(系)	计算机软件学院	专 业	软件工程

实 验 报 告

实验名称	俄罗斯方块			指导教师	张卫丰
实验类型	上机	实验学时	6	实验时间	2017.12.28

一、 实验目的和要求

要求：编写俄罗斯方块游戏，实现常用的俄罗斯方块功能。

1. 编程思路清晰，图像美观（满分 30 分）
2. 代码书写规范，关键部分有注释 （满分 10 分）
3. 功能实现全面 （满分 30 分）
4. 文档书写规范完整 （满分 20 分）
5. 代码按照要求的方式提交（满分 10 分）

二、实验环境(实验设备)

Windows 7、10

Visual Studio 2017

三、实验原理及内容

1.项目功能

本项目主要是实现俄罗斯方块基本功能。主要是游戏界面显示、开始结束、难度选择、计分、显示下一方块形状、满一行后消掉方块、方块的移动（左移、右移、加速向下、旋转）。如图 1 所示。

（1）显示游戏的界面

游戏玩家选择开始游戏打开游戏界面，便会显示游戏整个界面、游戏难度以及活动的方块和下一个方块。

（2）计分显示

根据游戏玩家在游戏当中的操作显示增加对应分数。

（3）难度选择功能

该功能实现俄罗斯方块游戏的难度数的增加，选择不同难度，界面显示不同的难度数，同时在游戏显示区显示的效果为方块下落的速度变化，使游戏的难度增加或减小。

（4）开始、结束游戏功能键

游戏玩家进入游戏界面之后点击游戏选择菜单下的开始游戏、结束游戏就能重新开始或直接结束俄罗斯方块游戏。

（5）显示下一方块形状

在游戏玩家进行游戏的过程中，实时显示下一个俄罗斯方块游戏的形状。

（6）满一行后消掉方块

在游戏玩家进行游戏的过程中，实时判断到达底部的方块是简单叠加还是引发消除事件。

（8）方块的移动（左移、右移、加速向下、旋转）

游戏玩家在游戏的过程中可以做的操作有：左移、右移、加速向下、旋转方块。

图 1 俄罗斯方块功能图

2. 原型界面

俄罗斯方块的原型图如图 2 所示。

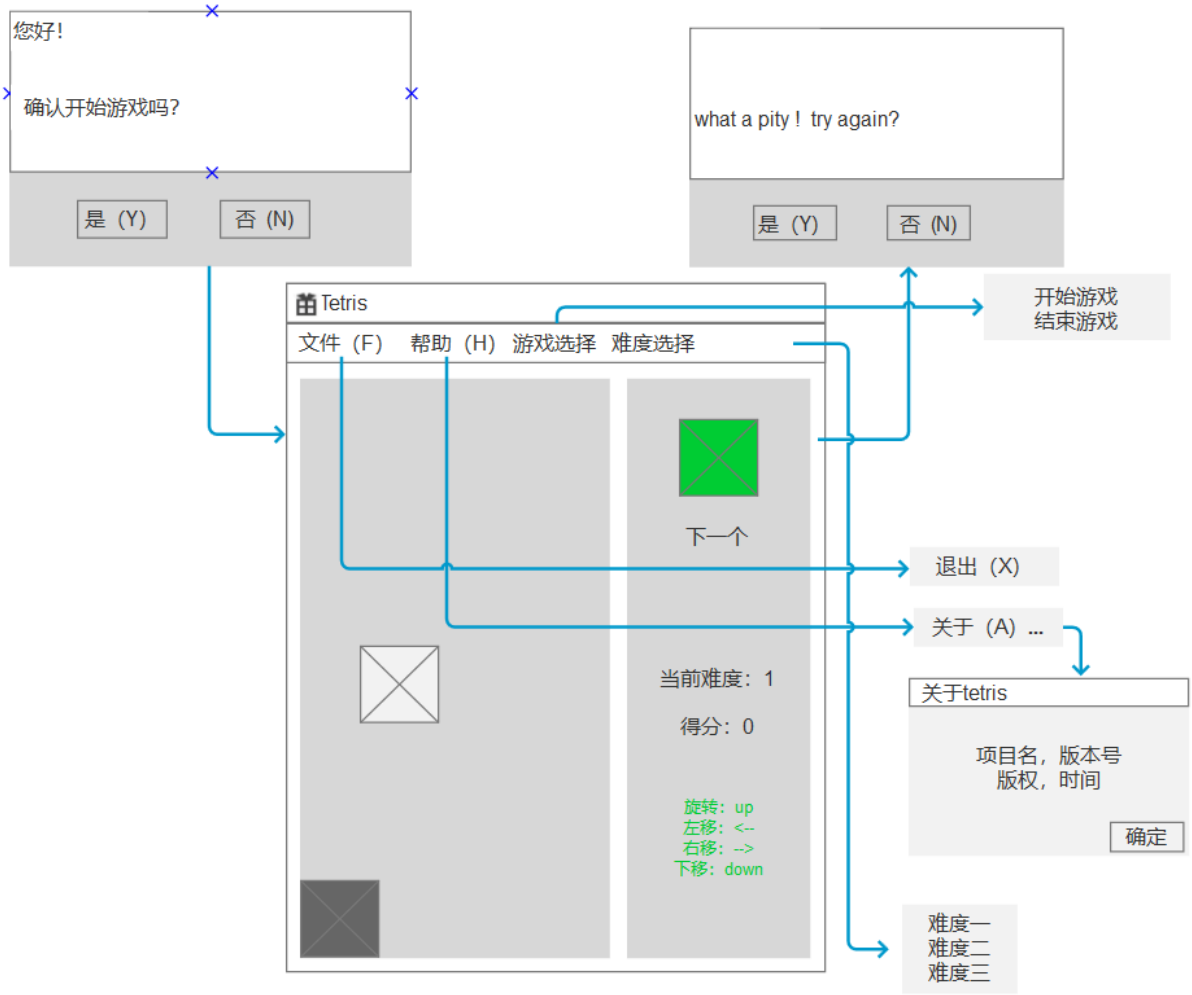

图 2 俄罗斯方块原型图

3. 数据结构与功能函数

本项目的主要数据结构和功能函数如图 3 和图 4 所示。

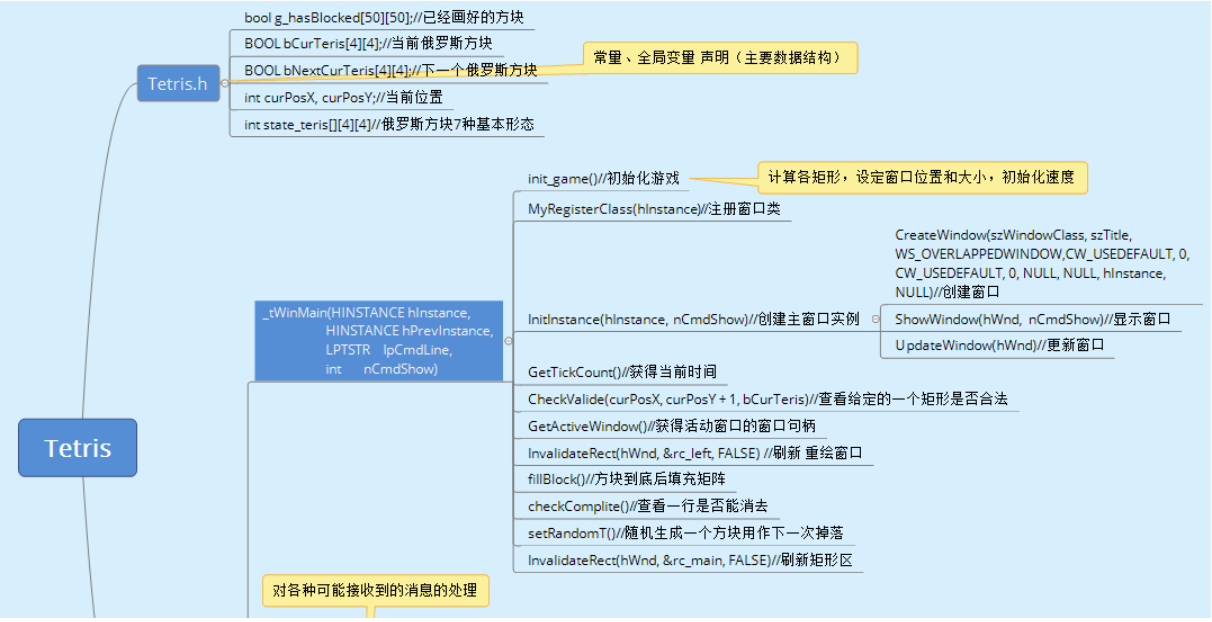

图 3 主要数据结构和功能函数图 1

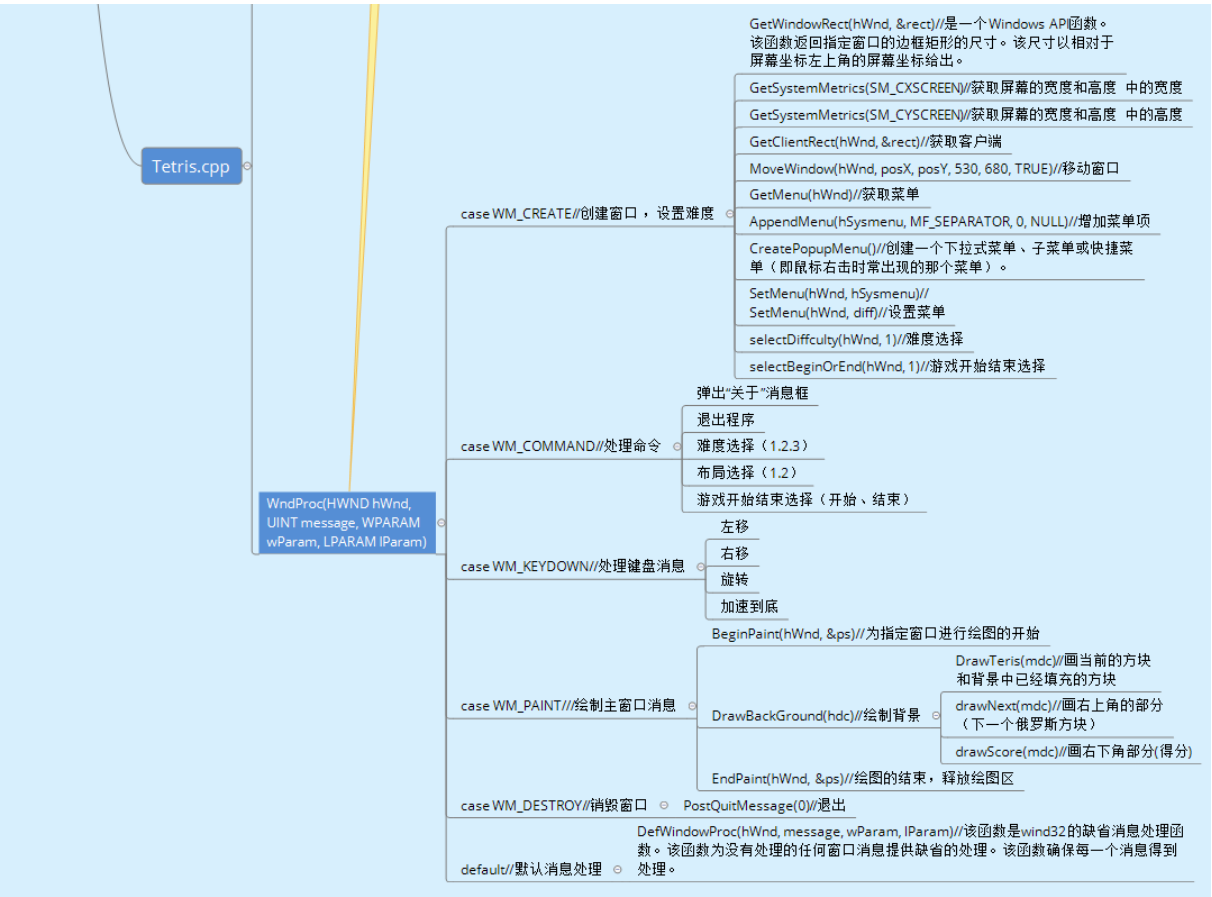

图 4 主要数据结构和功能函数图 2

4. 关键代码

4.1 主要数据结构

```
bool g_hasBlocked[50][50];//已经画好的方块
BOOL bCurTeris[4][4];//当前俄罗斯方块
BOOL bNextCurTeris[4][4];//下一个俄罗斯方块
int curPosX, curPosY;//当前位置
const BOOL state_teris[][4][4]//俄罗斯方块 7 种基本形态
```

4.2 主要功能函数

```
int CheckValide(int startX, int startY, BOOL bTemp[4][4])//给定一个矩阵查看是否合法
{
    int i, j;
    for (i = 3; i >= 0; i--)
    {
        for (j = 3; j >= 0; j--)
        {
            if (bTemp[i][j])
            {
                if (j + startX<0 || j + startX >= NUM_X)
                {
                    return -1;
                }
                if (i + startY >= NUM_Y)
                {
                    return -2;
                }
                if (g_hasBlocked[i + startY][j + startX])
                {
                    //outPutBoxInt(j+startY);
                    if (curPosY == 0)
                    {
                        return -3;
```

```
                        }
                        return -2;
                    }
                }
            }
        }
        //MessageBox(NULL,L"这里",L"as",MB_OK);
        //outPutBoxInt(curPosY);
        return 1;
}

void RotateTeris(BOOL bTeris[4][4])//方块旋转
{
        BOOL bNewTeris[4][4];
        int x, y;
        for (x = 0; x<4; x++)
        {
            for (y = 0; y<4; y++)
            {
                    //顺时针旋转90度：
                bNewTeris[x][y] = bTeris[3 - y][x];
                //逆时针：
                //bNewTeris[x][y] = bTeris[y][3-x];
            }
        }
        if (CheckValide(curPosX, curPosY, bNewTeris) == 1)
        {
            memcpy(bTeris, bNewTeris, sizeof(bNewTeris));//内存块的拷贝
        }
```

```
}

void checkComplite()//查看一行是否能消去  采用从上往下的消法，消去一行后把上面的每行都往下移
{
    int i, j, k, count = 0;
    for (i = 0; i<NUM_Y; i++)
    {
        bool flag = true;
        for (j = 0; j<NUM_X; j++)
        {
            if (!g_hasBlocked[i][j])
            {
                flag = false;
            }
        }
        if (flag)
        {
            count++;
            for (j = i; j >= 1; j--)
            {
                for (k = 0; k<NUM_X; k++)
                {
                    g_hasBlocked[j][k] = g_hasBlocked[j - 1][k];
                }
            }
            drawCompleteParticle(i);
            Sleep(300);
            PlaySound(_T("coin.wav"), NULL, SND_FILENAME | SND_ASYNC);
```

```
            }
        }
        GAME_SCORE += int(count*1.5);
}

void drawBlocked(HDC mdc)// 绘制当前已经存在砖块的区域
{
        int i, j;

        //HBRUSH hBrush = (HBRUSH)CreateSolidBrush(RGB(255, 255, 0));
        HBRUSH hBrush = (HBRUSH)CreateSolidBrush(RGB(51,51, 51));

        SelectObject(mdc, hBrush);

        for (i = 0; i<NUM_Y; i++)
        {
            for (j = 0; j<NUM_X; j++)
            {
                if (g_hasBlocked[i][j])
                {
                    Rectangle(mdc, BORDER_X + j*BLOCK_SIZE, BORDER_Y +
i*BLOCK_SIZE,
                        BORDER_X + (j + 1)*BLOCK_SIZE, BORDER_Y + (i +
1)*BLOCK_SIZE
                    );//画矩形
                }
            }
        }
        DeleteObject(hBrush);
```

```
}

VOID DrawBackGround(HDC hdc) //绘制背景
{

    HBRUSH hBrush = (HBRUSH)GetStockObject(GRAY_BRUSH);
    HDC mdc = CreateCompatibleDC(hdc);
    HBITMAP hBitmap = CreateCompatibleBitmap(hdc, SCREEN_X, SCREEN_Y);

    SelectObject(mdc, hBrush);
    SelectObject(mdc, hBitmap);

    HBRUSH hBrush2 = (HBRUSH)GetStockObject(WHITE_BRUSH);
    FillRect(mdc, &rc_main, hBrush2);//填充
    Rectangle(mdc, rc_left.left + BORDER_X, rc_left.top + BORDER_Y, rc_left.right,
rc_left.bottom);
    Rectangle(mdc, rc_right.left + BORDER_X, rc_right.top + BORDER_Y, rc_right.right,
rc_right.bottom);
    DrawTeris(mdc);
    drawNext(mdc);
    drawScore(mdc);
    ::BitBlt(hdc, 0, 0, SCREEN_X, SCREEN_Y, mdc, 0, 0, SRCCOPY);
    DeleteObject(hBrush);
    DeleteDC(mdc);
    DeleteObject(hBitmap);
    DeleteObject(hBrush2);
}

void setRandomT()//随机生成一个方块用作下一次掉落
```

```
{
    int rd_start = RandomInt(0, sizeof(state_teris) / sizeof(state_teris[0]));
    int rd_next = RandomInt(0, sizeof(state_teris) / sizeof(state_teris[0]));
    //outPutBoxInt(rd_start);
    //outPutBoxInt(rd_next);
    //outPutBoxInt(rd_start);
    if (GAME_STATE == 0)
    {
        GAME_STATE = GAME_STATE | 0x0001;
        //outPutBoxInt(GAME_STATE);
        memcpy(bCurTeris, state_teris[rd_start], sizeof(state_teris[rd_start]));
        memcpy(bNextCurTeris, state_teris[rd_next], sizeof(state_teris[rd_next]));
    }
    else
    {
        memcpy(bCurTeris, bNextCurTeris, sizeof(bNextCurTeris));
        memcpy(bNextCurTeris, state_teris[rd_next], sizeof(state_teris[rd_next]));
    }

}

void init_game()//初始化,里面保存程序的基本数据结构
{
    GAME_SCORE = 0;
    setRandomT();
    curPosX = (NUM_X - 4) >> 1;
    curPosY = 0;
```

```
    memset(g_hasBlocked, 0, sizeof(g_hasBlocked));
    rc_left.left = 0;
    rc_left.right = SCREEN_LEFT_X;
    rc_left.top = 0;
    rc_left.bottom = SCREEN_Y;

    rc_right.left = rc_left.right + BORDER_X;
    rc_right.right = 180 + rc_right.left;
    rc_right.top = 0;
    rc_right.bottom = SCREEN_Y;

    rc_main.left = 0;
    rc_main.right = SCREEN_X;
    rc_main.top = 0;
    rc_main.bottom = SCREEN_Y;

    rc_right_top.left = rc_right.left;
    rc_right_top.top = rc_right.top;
    rc_right_top.right = rc_right.right;
    rc_right_top.bottom = (rc_right.bottom) / 2;

    rc_right_bottom.left = rc_right.left;
    rc_right_bottom.top = rc_right_top.bottom + BORDER_Y;
    rc_right_bottom.right = rc_right.right;
    rc_right_bottom.bottom = rc_right.bottom;

    g_speed = t_speed = 1000 - GAME_DIFF * 280;
}
```

```
void fillBlock()//到达底部后填充矩阵
{
    int i, j;
    for (i = 0; i<4; i++)
    {
        for (j = 0; j<4; j++)
        {
            if (bCurTeris[i][j])
            {
                g_hasBlocked[curPosY + i][curPosX + j] = 1;
            }
        }
    }
}

VOID DrawTeris(HDC mdc) //绘制正在下落的方块
{

    int i, j;
    HPEN hPen = (HPEN)GetStockObject(BLACK_PEN);//画笔类型
    HBRUSH hBrush = (HBRUSH)GetStockObject(WHITE_BRUSH); //画刷类型
    SelectObject(mdc, hPen); //选择对应画笔
    SelectObject(mdc, hBrush); //选择对应画刷
    for (i = 0; i<4; i++)
    {
        for (j = 0; j<4; j++)
        {
            if (bCurTeris[i][j])
            {
```

```
                    Rectangle(mdc, (j + curPosX)*BLOCK_SIZE + BORDER_X, (i +
curPosY)*BLOCK_SIZE + BORDER_Y, (j + 1 + curPosX)*BLOCK_SIZE + BORDER_X,
(i + 1 + curPosY)*BLOCK_SIZE + BORDER_Y);
                }
            }
        }
        drawBlocked(mdc);
        DeleteObject(hPen);
        DeleteObject(hBrush);
}

void drawNext(HDC hdc) //绘制下一个将要掉落的方块
{
        int i, j;
        HBRUSH hBrush = (HBRUSH)CreateSolidBrush(RGB(0, 188, 0));
        SelectObject(hdc, hBrush);
        for (i = 0; i<4; i++)
        {
            for (j = 0; j<4; j++)
            {
                if (bNextCurTeris[i][j])
                {
                    Rectangle(hdc, rc_right_top.left + BLOCK_SIZE*(j + 1), rc_right_top.top +
BLOCK_SIZE*(i + 1), rc_right_top.left + BLOCK_SIZE*(j + 2), rc_right_top.top +
BLOCK_SIZE*(i + 2));
                }
            }
        }
        HFONT hFont = CreateFont(30, 0, 0, 0, FW_THIN, 0, 0, 0, UNICODE, 0, 0, 0, 0, L"微
```

```
软雅黑");
    SelectObject(hdc, hFont);
    SetBkMode(hdc, TRANSPARENT);
    SetBkColor(hdc, RGB(255, 255, 0));
    RECT rect;
    rect.left = rc_right_top.left + 60;
    rect.top = rc_right_top.bottom - 150;
    rect.right = rc_right_top.right;
    rect.bottom = rc_right_top.bottom;
    DrawTextW(hdc, TEXT("下一个"), _tcslen(TEXT("下一个")), &rect, 0);//显示文字

    DeleteObject(hFont);
    DeleteObject(hBrush);
}

void drawScore(HDC hdc) //绘制分数
{
    HFONT hFont = CreateFont(30, 0, 0, 0, FW_THIN, 0, 0, 0, UNICODE, 0, 0, 0, 0, L"微
软雅黑");
    SelectObject(hdc, hFont);
    SetBkMode(hdc, TRANSPARENT);
    SetBkColor(hdc, RGB(255, 255, 0));
    RECT rect;
    rect.left = rc_right_bottom.left;
    rect.top = rc_right_bottom.top;
    rect.right = rc_right_bottom.right;
    rect.bottom = rc_right_bottom.bottom;
    TCHAR szBuf[30];
    LPCTSTR cstr = TEXT("当前难度：%d");
```

```
    wsprintf(szBuf, cstr, GAME_DIFF);//格式化输出
    DrawTextW(hdc, szBuf, _tcslen(szBuf), &rect, DT_CENTER | DT_VCENTER);

    RECT rect2;
    rect2.left = rc_right_bottom.left;
    rect2.top = rc_right_bottom.bottom -230;
    rect2.right = rc_right_bottom.right;
    rect2.bottom = rc_right_bottom.bottom;
    TCHAR szBuf2[30];
    LPCTSTR cstr2 = TEXT("得分：%d");
    wsprintf(szBuf2, cstr2, GAME_SCORE);
    //outPutBoxInt(sizeof(szBuf));
    DrawTextW(hdc, szBuf2, _tcslen(szBuf2), &rect2, DT_CENTER | DT_VCENTER);

    HFONT hFont2 = CreateFont(23, 0, 0, 0, FW_THIN, 0, 0, 0, UNICODE, 0, 0, 0, 0, L"微软雅黑");
    SelectObject(hdc, hFont2);
    SetTextColor(hdc, RGB(0, 188, 0));
    SetBkMode(hdc, TRANSPARENT);//设置背景透明
    SetBkColor(hdc, RGB(255, 255, 0));

    RECT rect3;
    rect3.left = rc_right_bottom.left;
    rect3.top = rc_right_bottom.bottom-180;
    rect3.right = rc_right_bottom.right;
    rect3.bottom = rc_right_bottom.bottom;
    TCHAR szBuf3[30];
    LPCTSTR cstr3 = TEXT("旋转：up");
    wsprintf(szBuf3, cstr3, GAME_SCORE);
```

```
//outPutBoxInt(sizeof(szBuf));
DrawTextW(hdc, szBuf3, _tcslen(szBuf3), &rect3, DT_CENTER | DT_VCENTER);

RECT rect4;
rect4.left = rc_right_bottom.left;
rect4.top = rc_right_bottom.bottom - 150;
rect4.right = rc_right_bottom.right;
rect4.bottom = rc_right_bottom.bottom;
TCHAR szBuf4[30];
LPCTSTR cstr4 = TEXT("左移：<--");
wsprintf(szBuf4, cstr4, GAME_SCORE);
//outPutBoxInt(sizeof(szBuf));
DrawTextW(hdc, szBuf4, _tcslen(szBuf4), &rect4, DT_CENTER | DT_VCENTER);

RECT rect5;
rect5.left = rc_right_bottom.left;
rect5.top = rc_right_bottom.bottom - 120;
rect5.right = rc_right_bottom.right;
rect5.bottom = rc_right_bottom.bottom;
TCHAR szBuf5[30];
LPCTSTR cstr5 = TEXT("右移：-->");
wsprintf(szBuf5, cstr5, GAME_SCORE);
//outPutBoxInt(sizeof(szBuf));
DrawTextW(hdc, szBuf5, _tcslen(szBuf5), &rect5, DT_CENTER | DT_VCENTER);

RECT rect6;
rect6.left = rc_right_bottom.left;
rect6.top = rc_right_bottom.bottom - 90;
rect6.right = rc_right_bottom.right;
```

```
    rect6.bottom = rc_right_bottom.bottom;
    TCHAR szBuf6[30];
    LPCTSTR cstr6 = TEXT("下移：down");
    wsprintf(szBuf6, cstr6, GAME_SCORE);
    //outPutBoxInt(sizeof(szBuf));
    DrawTextW(hdc, szBuf6, _tcslen(szBuf6), &rect6, DT_CENTER | DT_VCENTER);

    DeleteObject(hFont);
}

int selectDiffculty(HWND hWnd, int diff)//选择难度
{
    TCHAR szBuf2[30];
    LPCTSTR cstr2 = TEXT("确认选择难度 %d 吗？");
    wsprintf(szBuf2, cstr2, diff);
    if (MessageBox(hWnd, szBuf2, L"难度选择", MB_YESNO) == IDYES)
    {
        GAME_DIFF = diff;
        InvalidateRect(hWnd, &rc_right_bottom, false);//使无效,刷新矩形区域
        GAME_STATE |= 2;
        //init_game();
        g_speed = t_speed = 1000 - GAME_DIFF * 280;
        return GAME_DIFF;
    }
    return -1;
}
```

5. 实验结果

5.1 开始游戏功能实现

游戏开始的界面，玩家只需点击“是”来确认即可开始游戏（或者点击“否”退出

游戏）。俄罗斯方块游戏开始界面如图 5 所示。

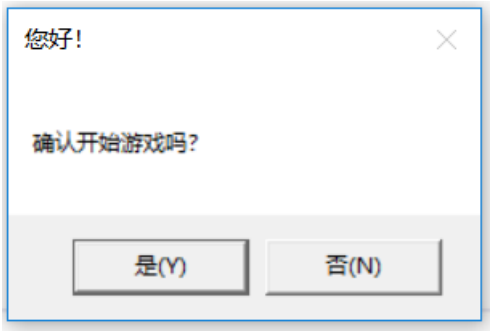

图 5 游戏开始界面图

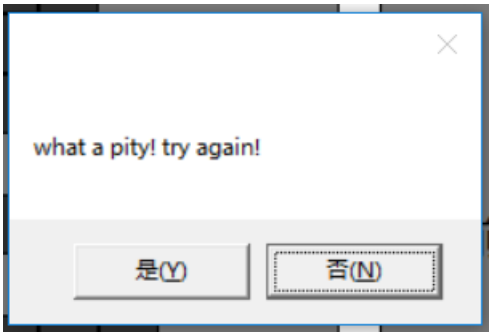

图 7 游戏结束界面图

5.2 游戏界面功能实现

玩家确认开始游戏后便进入游戏界面。玩家在游戏界面可以上下左右键对方块进行操作来使得方块尽可能的消去从而计分。玩家在游戏界面也可以选择游戏的难度、重新开始画或退出游戏等。俄罗斯方块游戏界面如图 6 所示。

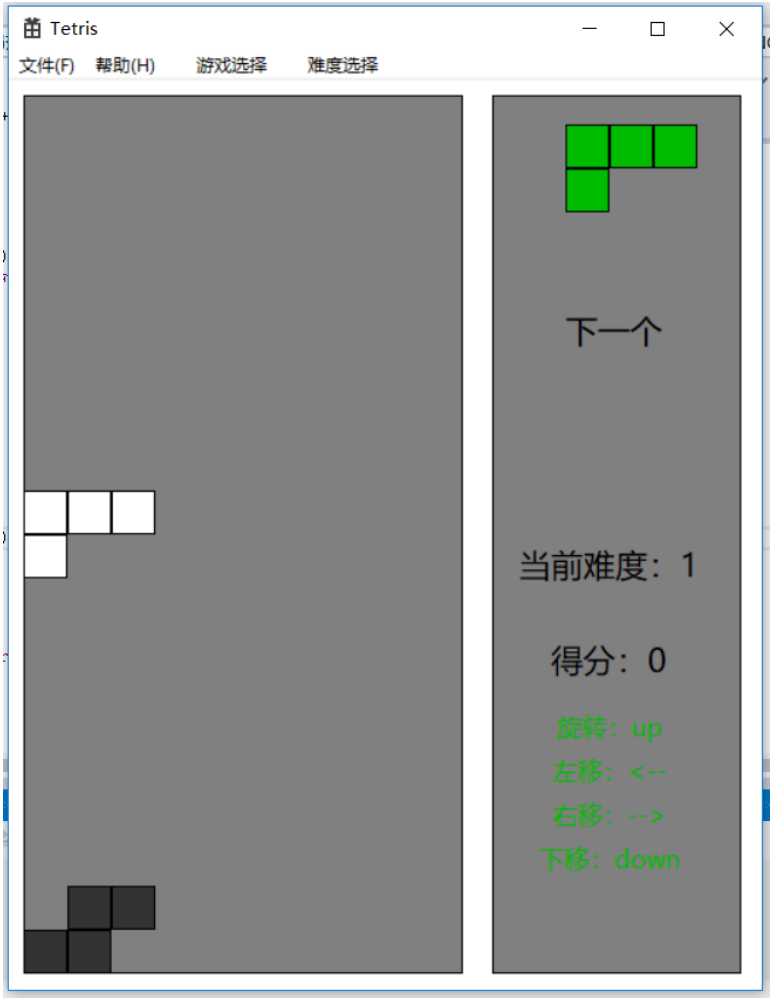

图 6 游戏界面图

5.3 游戏结束功能实现

如果玩家未能在规定行数内完成消去任务，游戏便结束了，此时弹出一个弹框，玩家可以选择再来一次或者退出游戏。俄罗斯方块游戏失败结束界面如图 7 所示。

四、实验小结

1. 遇到的问题及解决方案

（1） 进行 GTEST 时候，花了很多时间。需要注意事项列出如下：

1）GTEST 和项目要配置一致：都是 x64 或者都是其他的如 x86 等。

2）GTEST 测试时，下面图 8 蓝色选项要选择“控制台”而不是“Windows”，否则会报错“error LNK2019: 无法解析的外部符号 _main”。

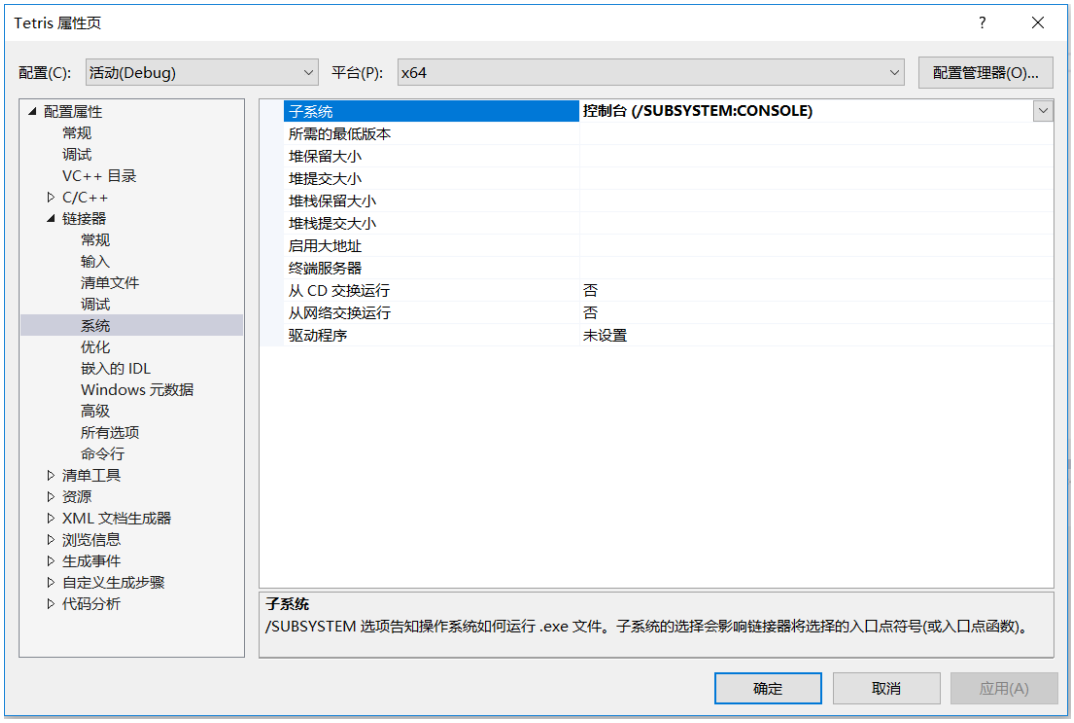

图 8 项目属性设置图

（2） 加入数据结构报错：未定义标识符 BOOL RECT

解决：加入#include <windows.h>这个头文件即可，因为他们这两个关键字是在这个头文件里的。

（3） 加入 draw*函数时报错：找不到标识符

解决：因为没有在.h 文件中加上函数声明，加上即可。

2. 总结体会

本次实验，对于自身编程水平是个不小的提高。之前，很少动手。而这次实验，几乎都是自己动手完成的，有些借鉴的地方，也都自己查阅资料进行错误调试改正。此次实验，让我深刻感受到了实践出真知的道理。因此，在以后的学习中，要树立实践的意

识，提高把理论知识上升为实践的能力。

在本次实验的开始阶段，老师一直强调一个项目的框架、整体要实现那些功能。可见，只有对一个项目进行合理可行的分析与规划，才能使得后续项目顺利的进行。因此在以后的实验以及学习中，在任务正式开始之前，要充分合理的对项目进行可行的规划，避免一团乱麻的现象出现。

本次实验之前，研究了一些有关俄罗斯方块的例子，体悟它们程序实现的方法，并加以借鉴。在阅读他人程序时候，多多少少所有不懂的地方，这个时候，书本以及网络是非常好的工具。适当的借鉴和对案列的充分分析加上书本互联网工具的充分使用，才能把一个项目做好。有一句话叫“站在巨人的肩膀上看”，就是此次实验给我的感觉。也就是，在理解前人的思路的基础上加上自己的思想，也许会有新发现。

分部编写各个函数，并对其逐一进行测试，化大为小，是本次实践的一大收获，是编写较复杂程序的有效方法。这样有利于条理的清晰，避免错误。

合作与交流是成功的捷径。与同课题的同学的交流，让我产生思想的火花，在困惑自己许久的问题上带来突破。

最后，本次实验非常感谢老师的指导与耐心答疑，使得我在此次实验上收获很大，如 GTEST 问题的解决等。本次实验掌握了很多技能如 gitee 及其客户端 github 的熟练使用，xmind 以及 Axure 的使用等，使得我在未来的学习之路上有了很好的突破。综上，本次实验收益匪浅。

五、指导教师评语

成　绩		批阅人		日　期	