无标题

浙江大学

程 序 设 计 专 题

大 程 序 报 告

1. 学生姓名 ： 施楚宁 学号： 3180102763

2. 学生姓名 ： 王逸旦 学号： 3180103818

3. 学生姓名 ： 徐 珂 学号： 3180103434

指导老师： 陈建海

2018~2019春夏学期 2019 年 5 月 31 日

报告撰写注意事项

1. 图文并茂。文字通顺，语言流畅，无错别字。
2. 书写格式规范，排版良好，内容完整。
3. 存在拼凑、剽窃等现象一律认定为抄袭；0分

1. 题目描述和题目要求

1.1 题目描述

俄罗斯方块是一款风靡全球的经典游戏，是不少人的童年记忆。它由俄罗斯人阿列克谢·帕基特诺夫发明，故得此名。该游戏通过移动、旋转和摆放游戏自动输出的各种方块，使之排列成完整的一行或多行并消除得分。

1.2 题目要求

基于libgraphics，设计和实现俄罗斯方块。程序必须用多文件组织的方式实现，应当支持键盘操作，通过不同的功能键实现方块形状、位置、下落速度的变换，设置游戏菜单、排行榜等，以实现俄罗斯方块游戏的基本功能。需要运用链表、文件等知识，实现系统数据的组织和存储。设计过程中要尽量遵守代码规范，优化代码执行速度，并尽量使最终的游戏界面简洁美观、简单易用。

2. 需求分析

俄罗斯方块游戏需要解决的问题

（1）绘制游戏界面，生成标准的用于摆放俄罗斯方块的10×20区域；

（2）实现方块的随机生成与自动下落；

（3）实现不同的按键功能：

↑：90°旋转正在下落的方块；在游戏暂停时可以上调玩家的等级；

↓：使方块加速下落；在游戏暂停时可以下调玩家的等级；

←/→：使正在下落的方块左移/右移；

空格键：使正在下落的方块直接落到底部；

ESC键：暂停游戏，暂停后可选择继续游戏或直接退出游戏；

x键：实现俄罗斯方块的“Hold”功能。当“Hold”预览窗为空时，按”x”键可以将”Next One”预览窗中的方块保留至”Hold“区域，并刷新”Next One”预览窗；当”Hold“预览窗非空时，按”x”键可以使”Hold”预览窗中的俄罗斯方块成为下一个下落的俄罗斯方块,并清空“Hold”预览窗。

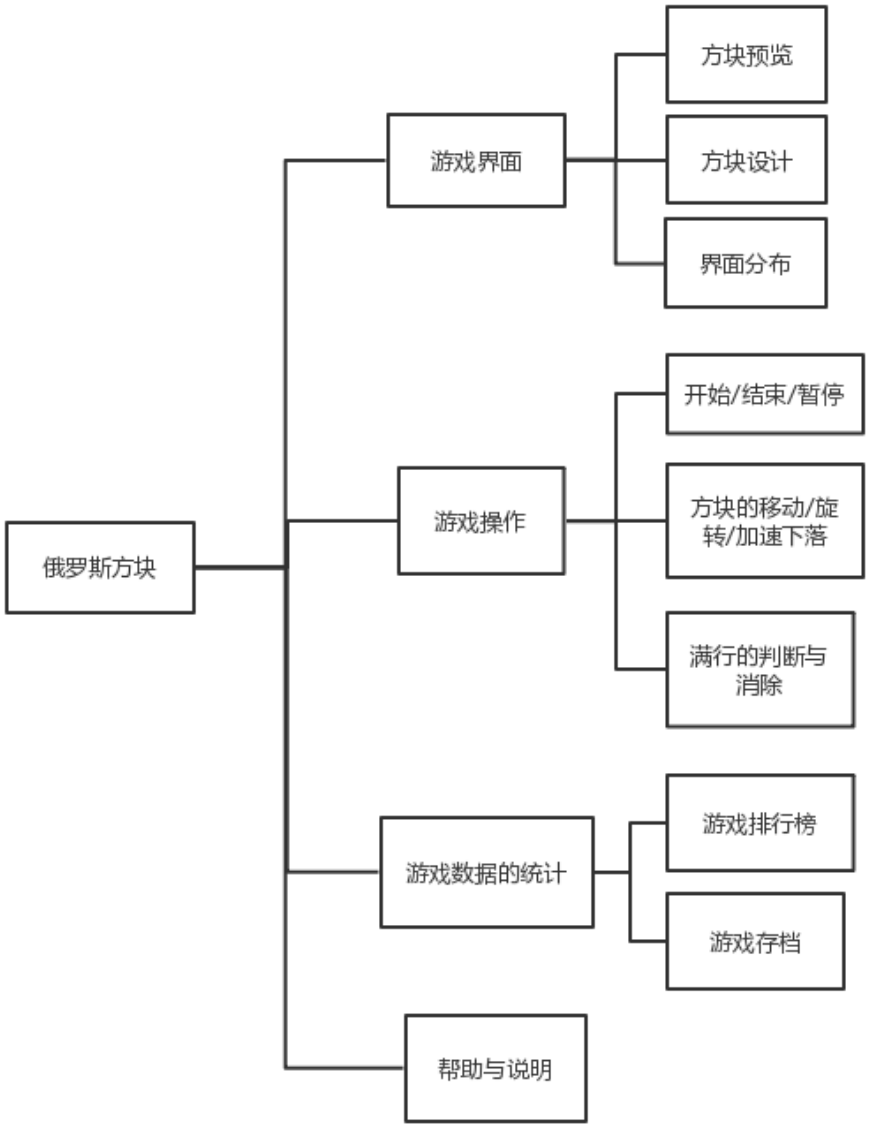
（4）能正确判定行是否填满，并消除满行，按一定的规则累加积分，设定级别；

（5）游戏结束后，储存玩家姓名和得分并对玩家进行排名；

（6）实现游戏存档功能。

3.总体设计

3.1 功能模块设计



3.2数据结构设计

在本程序中，每个俄罗斯方块都用结构体来表示，在DrawUtils.h中利用typdef将俄罗斯方块所对应的结构体类型名定义为Tetromino，结构体定义如下：

```
12  typedef struct {
13      int x, y;
14      int direction;
15      int type;
16  } Tetromino;
```

其中，整型变量x,y用于表示方块所处的位置，整型变量direction用来表示方块当前的朝向，整型变量type用来表示方块所属的类型。

俄罗斯方块游戏中，一共有7种类型的方块。在本程序中，我们用1~7分别对应这七种方块，数字0无对应的方块，不同类型的方块对应着不同的颜色。对应关系如下：
0 – 无 – 无

1 – 浅蓝 (Cyan) – □□□□ – I型方块

2 – 蓝 (Blue) – □□□ – L型方块

3 – 橙 (Orange) – □□□ – J型方块

4 – 黄 (Yellow) – □□ – O型方块

5 – 绿 (Green) – □□ – S型方块

6 – 紫 (Violet) – □□□ – T型方块

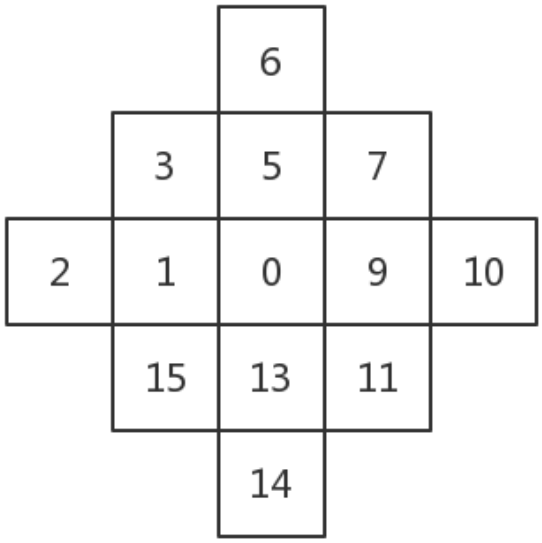
7 – 红 (Red) – □□ – Z型方块

可以看到，每种俄罗斯方块都是由4个小正方形组成的。为了表示这几种方块，我们又在DrawUtils.c中规定了名为TetrominoShape的8×4的整型二维数组：

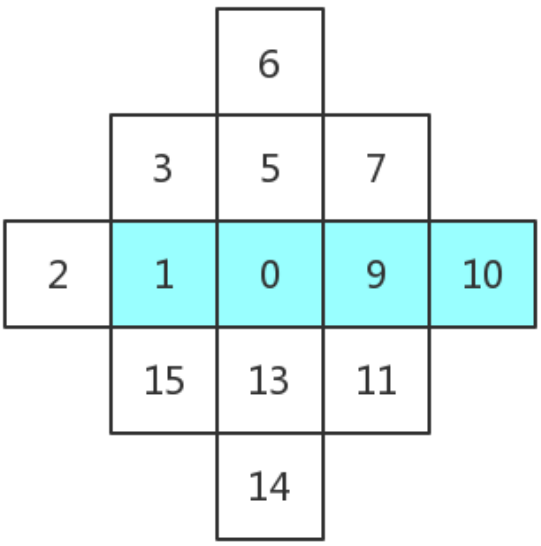
```
57  int TetrominoShape[8][4] = {
58          {0, 0, 0,  0},
59          {0, 1, 9,  10},
60          {0, 1, 3,  9},
61          {0, 1, 9,  7},
62          {0, 9, 11, 13},
63          {0, 1, 5,  7},
64          {0, 1, 5,  9},
65          {0, 3, 5,  9}
66  };
```

该二维数组的8个行下标分别对应着俄罗斯方块的类型（其中0对应的为空）。对某一确定类型的俄罗斯方块而言，结构体Tetromino中的type的取值与TetrominoShape[8][4]数组的行下标对应相等。拥有相同行下标的数组元素可以视为一个数列，数列元素的取值为大于等于0，小于等于15，且非4的倍数的正整数。这些正整数按照下图的方式排列，数列元素取到某个值着图中对应位置的方块被填充，这些被填充的方块一起组成了某一类型的俄罗斯方块。可以观察到图中从0所在的中心位置出发，向各个数字所在的方块连线。长度相等且夹角为90°

的连线所对应的终点位置的数字模4同余，这是考虑到俄罗斯方块每次旋转90°，旋转四次即



可回到原来的位置。：
以行下标为1的元素为例，它们对应的值分别为1、0、9、10，也即这些数字所在的小正方形是被填充了的，对应的图案就是水平放置的I形方块。



通过以上设计，我们将俄罗斯方块的形状用数字的形式储存在了数组中。接下来我们还应当设计出一种计算方法对数组元素进行变换，使得变换后的元素能对应旋转后的俄罗斯方块。
于是我们又设计了二维数组int Direction[16][2]。

```
int Direction[16][2] = {
        {0,  0},
        {-1, 0},
        {-2, 0},
        {-1, 1},
        {0,  0},
        {0,  1},
        {0,  2},
        {1,  1},
        {0,  0},
        {1,  0},
        {2,  0},
        {1,  -1},
        {0,  0},
        {0,  -1},
        {0,  -2},
        {-1, -1}
};
```

在函数DrawTetromino中，可以看到二维数组Direction[16][2]中的元素参与了俄罗斯方块的x、y坐标变换，实现了俄罗斯方块的旋转。

```
void DrawTetromino(Tetromino x) {
    int dir = x.direction % TetrominoDirectionMod[x.type];
    for (int i = 0; i < 4; i++) {
        DrawBlocks(x.x + Direction[(TetrominoShape[x.type][i] + 4 * dir) % 16][0],
                x.y + Direction[(TetrominoShape[x.type][i] + 4 * dir) % 16][1],
                1, 1, TetrominoColor[x.type],
                DEFAULT_COLOR);
    }
}
```

为了简化运算，我们又引入了整型数组TetrominoDirectionMod[8]。由于数字0所对应的空方块和数字5所对那个的O型方块经历任意次变换都与原状相同，所以TetrominoDirection[0]与TetrominoDirection[5]的取值为1。数字1所对应的I型方块经历2次变换后就与原状相同，所以TetrominoDirection[1]的取值为2。同理，数组中其余的元素取值为4。

```
68  int TetrominoDirectionMod[8] = {
69      1, 2, 4, 4, 1, 4, 4, 4
70  };
```

3.3 函数功能描述

3.3.1 DrawUtils

1.

函数原型：void DrawTetromino(Tetromino x)
功能描述：根据输入的x，绘制出当前下落的俄罗斯方块
参数描述： Tetromino x表示当前下落的俄罗斯方块
返回值描述：无返回值
重要局部变量定义：int dir = x.direction % TetrominoDirectionMod[x.type]
重要局部变量用途描述：利用设定好的数组元素简化计算，详情可见3.2
函数算法描述：调用函数DrawBlocks，利用for循环逐一绘制俄罗斯方块中的4个小方块。

2.

函数原型：void DrawTetrominoOutline(Tetromino x)
功能描述：根据输入的X的参数，画出具体的下落的一种俄罗斯方块的轮廓线
参数描述： Teromino x 表示当前下落的俄罗斯方块
返回值描述：无返回值
重要局部变量定义：int dir = x.direction % TetrominoDirectionMod[x.type]
重要局部变量用途描述：利用设定好的数组元素简化计算，详情可见3.2
函数算法描述：调用函数DrawOutline，利用for循环逐一绘制俄罗斯方块中的4个小方块。

3.

函数原型：void DrawFrame(double x, double y)
功能描述：以输入的（x,y)确认新的坐标原点以绘制游戏的整体框架,(x,y)默认为（0，0）
参数描述： x,y
返回值描述：无返回值
重要局部变量定义：int column[] = {0, 7, 18, 25}; char *ScoreText; char *LevelText
重要局部变量用途描述：column[]中的元素指明了游戏界面中不同区域的边界宽度，字符指针*ScoreText和*LevelText分别指向储存玩家得分和等级的字符串的地址
函数算法描述：调用drawBox,DrawBlocks,DrawMenu等函数绘制游戏框架中的不同组件。

4.

函数原型：void DrawMenu()
功能描述：绘制游戏菜单
参数描述： 无
返回值描述：无返回值
重要局部变量定义： static char *menuListFile[];static char *menuListTool[];static char *menuListHelp[];double x,y,w,h,wlist

重要局部变量用途描述：字符指针数组*menuListFile[]，*menuListTool[]，*menuListHelp中的元素分别指向了File、Tool、Help菜单下各个子选项卡的标题字符串的地址。变量x,y用于标明菜单的坐标，w标示了菜单的总宽度，h标示了菜单的高度，wlist标示了子菜单的宽度。

函数算法描述：

5.

函数原型：void DrawBlocks(int x, int y, int r, int c, char *InnerColor, char *OuterColor)

功能描述：根据输入参数，在确定的位置绘出确定颜色的方块，xyrc为确认在何位置画出多少数量的方块，字符串变量则以确定颜色

参数描述： x, y, r, c, InnerColor, OuterColor

返回值描述：无返回值

重要局部变量定义：

重要局部变量用途描述：变量x,y用于确定画方块的坐标，变量r,c则用于确定画出方块的数量， InnerColor, OuterColor确定了画出方块的外围颜色和填充色

6.

函数原型：void DrawOutline(int x, int y, int r, int c, char *Color)

功能描述：绘制图形的轮廓线，可参考DrawBlocks的注释

参数描述： x, y, r, c, Color

返回值描述：无返回值

重要局部变量定义：无局部变量

重要局部变量用途描述：参考DrawBlocks的注释

7.

函数原型：void DrawLayers(int head[12][22])

功能描述：调用函数进行行初始化后重新绘制整体的格子

参数描述： head[12][22]

返回值描述：无返回值

8.

函数原型：void DrawGameOver()

功能描述：当游戏结束时，提示玩家游戏结束并输入姓名，并弹出“Click to Retey”字样提示玩家单击即可重新运行游戏

返回值描述：无返回值

重要局部变量定义：无局部变量

9.

函数原型：void DrawRankList()

功能描述：绘制排行榜，并在左边显示玩家名字，右边显示玩家得分

参数描述： 本次游戏的得分currentNode->score及玩家输入的名字currentNode->name

返回值描述：无返回值

重要局部变量定义：char RankScoreText[10]，currentNode

重要局部变量用途描述：

RankScoreText[10]储存当前节点储存的成绩所转换的字符串

currentNode用于遍历排行榜的链表

10.

函数原型：void DrawGamePause()

功能描述：当玩家按下ESC时，游戏暂停，弹出重新开始及退出两个选项

返回值描述：无返回值

重要局部变量定义：无局部变量

11.

函数原型：void SetDefaultStyle()
功能描述：为按钮以及内置文字的方块设置初始颜色，并预置内部填充
返回值描述：无返回值
重要局部变量定义：无局部变量

12.

函数原型：void DrawInitPage()
功能描述：当游戏结束时，提示玩家游戏结束并输入姓名，并弹出“Click to Retey”字样提示玩家单击即可重新运行游戏
参数描述： 无
返回值描述：无返回值
重要局部变量定义：Title[7][35]
重要局部变量用途描述：用于储存开始页面的‘TETRIS’图标

13.

函数原型：void DrawClearBlink(int Clear[],int n)
功能描述：使即将被清除的满行闪烁
参数描述： Clear[]用于储存需要消除的行，n表示需要清除的总行数
返回值描述：无返回值
重要局部变量定义：无

14.

函数原型：void DrawAbout()
功能描述：绘制游戏菜单中的About栏
参数描述： 无
返回值描述：无返回值
重要局部变量定义：无

15.

函数原型：void DrawHelp()
功能描述：绘制游戏菜单中的Help栏，以及显示help栏中的内容
参数描述： 无
返回值描述：无返回值
重要局部变量定义：无

16.

函数原型：void DrawStatusBar()
功能描述：绘制游戏中的状态栏
参数描述： 无
返回值描述：无返回值
重要局部变量定义：无

3.3.2 GameUtils

1.

函数原型：void KeyboardEventProcess(int key, int event)
功能描述：每当产生键盘消息时，执行对应的功能，对应俄罗斯方块的各项游戏功能
参数描述： 由libgraphics库定义
返回值描述：无返回值

2.

函数原型：void MouseEventProcess(int x, int y, int button, int event)
功能描述：每当产生鼠标消息时，执行对应的功能

参数描述：由libgraphics库定义

返回值描述：无返回值

3.

函数原型：void TimerEventProcess(int timerID)

功能描述：每当产生计时器消息时，执行对应的功能

参数描述：由libgraphics库定义

返回值描述：无返回值

4.

函数原型：void CharEventProcess(char ch)

功能描述：每当产生字符串消息时，执行对应的功能

参数描述：由libgraphics库定义

返回值描述：无返回值

5.

函数原型：void RefreshCurrent()

功能描述：生成新的方块

返回值描述：无返回值

重要局部变量定义：无局部变量

6.

函数原型：void RefreshDisplay()

功能描述：清除上次游戏排列的方块，重新绘出游戏界面

返回值描述：无返回值

重要局部变量定义：无局部变量

7.

函数原型：void NewRound()

功能描述：初始化参数，开始新的一轮游戏

返回值描述：无返回值

重要局部变量定义：

重要局部变量用途描述：用于初始化记录名字的数组

函数算法描述：

8.

函数原型：void DrawResult()

功能描述：当游戏结束时执行相对应的功能

返回值描述：无返回值

重要局部变量定义：tmp

重要局部变量用途描述：用于记录current方块下落至底后的状态

9.

函数原型：void UpdateRank()

功能描述：若无排名则新建，若有排名则更新

返回值描述：无返回值

重要局部变量定义：无局部变量

10.

函数原型：void SwitchHold()

功能描述：实现游戏中的”Hold”功能，当“Hold”预览窗为空时，将”Next One”预览窗中的方块保留至“Hold”区域，并刷新”Next One”预览窗；当”Hold“预览窗非空时，使”Hold”预览窗中的俄罗斯方块成为下一个下落的俄罗斯方块，并清空“Hold”预览窗。

返回值描述：无返回值

重要局部变量定义：无局部变量

11.

函数原型：void GameOver()
功能描述：调用函数，完成当次游戏结束时的各项功能
返回值描述：无返回值
重要局部变量定义：无局部变量

12.

函数原型：void GamePause()
功能描述：暂停游戏
返回值描述：无返回值
重要局部变量定义：无局部变量

13.

函数原型：void GameResume()
功能描述：重新开始游戏
返回值描述：无返回值
重要局部变量定义：无局部变量

14.

函数原型：void GameExit(int save)
功能描述：保存本次游戏内容并退出游戏界面
参数描述：save表示退出游戏时是否需要保存当前游戏信息
返回值描述：无返回值
重要局部变量定义：无局部变量

15.

函数原型：void GameContinue()
功能描述：继续被暂停了的游戏
返回值描述：无返回值
重要局部变量定义：无局部变量

16.

函数原型：void PauseTimer()
功能描述：使计时器NORMAL_DOWN和ACCELRATE暂停工作
返回值描述：无返回值
重要局部变量定义：无局部变量

17.

函数原型：void ResumeTimer()
功能描述：使计时器NORMAL_DOWN重新开始工作
返回值描述：无返回值
重要局部变量定义：无局部变量

18.

函数原型：void UpdateLevel(int count)
功能描述：计算并更新玩家当前的等级
返回值描述：无返回值
重要局部变量定义：无局部变量

19.

函数原型：void ResetDownTimer()
功能描述：根据玩家当前的等级调整方块下落的速度并重置NORMAL_DOWN计时器
返回值描述：无返回值
重要局部变量定义：DownSpeed
重要局部变量用途描述：记录当前level的对应时间间隔
函数算法描述：

20.

函数原型：void LogStatusBar(MessageTypes MessageType)
功能描述：将状态栏更新为enum类的MessageType 对应的log信息
返回值描述：无返回值
重要局部变量定义：无局部变量
重要局部变量用途描述：
函数算法描述：

3.3.3 JudgeUtils

1.

函数原型：void LayerInit()

功能描述：行初始化

返回值描述：无返回值

重要局部变量定义：TetrominoMap[11][21]

重要局部变量用途描述：TetrominoMap[11][21]中的元素用来表示游戏界面的填充情况。对于数组元素TetrominoMap[i][j]，i表示对应方格的x坐标，j表示y坐标，TetrominoMap[i][j]==0则表示对应格未被填充，TetrominoMap[i][j]==1则表示对应格被填充。

函数算法描述：

2.

函数原型：void UpdateLayers(Tetromino x)
功能描述：更新方块x掉落后各行的状态，并调用函数ClearFullLayers以消除已填满的行
参数描述：Tetromino x表示正在下落的俄罗斯方块

返回值描述：无返回值

重要局部变量定义：dir

重要局部变量用途描述：简化当前方块的方向

函数算法描述：

3.

函数原型：void JudgeBorder(Tetromino x,int FallDirection)

参数描述：Tetromino x表示正在下落的俄罗斯方块；int FallDirection

功能描述：判断方块是否处于游戏界面边界

返回值描述：返回0代表方块不处于边界，返回1代表方块触及了边界

重要局部变量定义：dir

重要局部变量用途描述：简化当前方块的方向

函数算法描述：

4.

函数原型：void ClearFullLayer()

功能描述：清除已填满的行并将上方的方块对应下移并给玩家加上对应的分数

返回值描述：无返回值

重要局部变量定义：count

重要局部变量用途描述：记录需要消除的行数

函数算法描述：

5.

函数原型：void JudgeGameOver(Tetromino x)

参数描述：Tetromino x应当赋值为当前下落的俄罗斯方块

功能描述：判断当前俄罗斯方块下落后，方块堆积的高度是否超过了游戏界面的高度，从而判断游戏是否结束

返回值描述：返回0代表游戏可以继续，返回1代表方块触及了边界，游戏结束

重要局部变量定义：dir

重要局部变量用途描述：简化当前方块的方向

3.3.4 ListUtils

1.

函数原型：ListNodePtr CreateList(void)

功能描述：创建链表并返回头结点

返回值描述：ListNodePtr head

重要局部变量定义：ListNodePtr head

重要局部变量用途描述：以ListNodePtr head为头结点创建链表

2.

函数原型：void freeList(ListNodePtr head)

参数描述：排行榜链表头结点 ListNodePtr head

功能描述：清空以head为头结点的链表

重要局部变量定义：ListNodePtr current

重要局部变量用途描述：current用于遍历怕行榜链表。

函数算法描述：首先将current赋值为head，当current对应的指针非空指针（NULL）时，将current赋值为NULL并后移current，直至current对应的指针为NULL，而current之前的数据都被清空。

3.

函数原型：void InsertNode(ListNodePtr head, int val, char* name)

参数描述：排行榜链表头结点ListNodePtr head，新的用户名char *name，新的用户得分int val

功能描述：将新的用户名和用户得分按得分排行插入到原有的链表中

返回值描述：返回储存了排行榜数据的链表的头结点head

重要局部变量定义：ListNodePtr current，previous

重要局部变量用途描述：current遍历排行榜链表，previous记录current的上一个节点

函数算法描述：遍历排行榜，找到第一个score属性大于等于val的节点，新建节点，其name属性赋值name，score属性赋值val，插入至previous后，current前

4.

函数原型：void DeleteNode(ListNodePtr head)

参数描述：排行榜链表头结点 ListNodePtr head

功能描述：当玩家数量超过了排行榜的最大容纳量时，删除排行榜上得分最低的玩家，并返回储存了排行信息的链表的头节点

返回值描述：返回储存了排行榜数据的链表的头结点head

重要局部变量定义：ListNodePtr current，int count

重要局部变量用途描述：current用于遍历排行榜链表；count用于记录链表元素个数

函数算法描述：当count等于10时将对应节点的next设为NULL

3.3.5 SaveUtils

1.

函数原型：void SaveGame()

功能描述：将游戏存档

返回值描述：无返回值

重要局部变量定义：FILE *data

重要局部变量用途描述：记录以wb模式打开存档文件的指针

函数算法描述：将关键变量写入至data

2.

函数原型：void RecoverGame()

功能描述：读取之前的游戏存档

返回值描述：无返回值

重要局部变量定义：FILE *data

重要局部变量用途描述：记录以rb模式打开存档文件的指针

函数算法描述：从data中读取关键变量

3.4 程序文件结构

3.4.1 文件函数结构

main.c：包含了主函数void Main()

DrawUtils.c：定义了函数：

```
void DrawTetromino(Tetromino x)
void DrawTetrominoOutline(Tetromino x)
void DrawFrame(double x, double y)
void DrawMenu()
void DrawBlocks(int x, int y, int r, int c, char *InnerColor, char *OuterColor)
void DrawOutline(int x, int y, int r, int c, char *Color)
void DrawLayers(int head[12][22])
void DrawGameOver()
void DrawRankList()
void DrawGamePause()
void SetDefaultStyle()
void DrawInitPage()
void DrawClearBlink(int Clear[], int n)
void DrawAbout()
void Help()
```

DrawUtils.h：

1）声明了DrawUtils.c中所定义的函数；

2）声明了双精度常量BlockLength，该变量用于表示方格的长度；

3）定义了分别名为DoublePoint和Tetromino的结构体类型；

4）声明了数组int Direction[16][2]，int TetrominoShape[8][4]和int TetrominoDirectionMod[8];

GameUtils.c：定义了函数：

```
void KeyboardEventProcess(int key, int event)
void MouseEventProcess(int x, int y, int button, int event)
void CharEventProcess(char ch)
void TimerEventProcess(int timerID)
void NewRound()
void UpdateRank()
void GamePause()
void GameResume()
void GameContinue()
void GameExit(int save)
void PauseTimer()
void ResumeTimer()
void UpdateLevel(int count)
void ResetDownTimer()
void RefreshDisplay()
```

GameUtils.h：

1）声明了GameUtils.c中所定义的函数；

2）声明了双精度变量DownSpeed；

3）声明了Tetromino型的结构体current，next，hold，分别用来表示正在下落的方块，即将下落的方块和“Hold”预览窗中的方块；

4）声明了用于储存玩家姓名的字符串数组char Name[NAMELENGTH]；

5）声明了用于储存方格填充情况的整型数组TetrominoMap[12][22]，用于储存加分梯度的整形数组ScoreAdd[5]；

6）声明了用于储存排行榜数据的链表ListNodePtr RankList；

7）声明了一些列标记程序当前状态的整型变量：

IsStop – 游戏是否停止
IsPause – 游戏是否暂停
CanHold – 当前能否将方块移到Hold预览窗
InitPage – 是否需要初始化界面
CanContinue – 游戏是否能继续
AllClearedLayer – 统计已经消除的总行数
Score – 玩家当前得分
Level – 玩家当前等级
int ShowAbout – 是否能展示About界面
int ShowHelp – 是否能展示Help界面

8）利用宏定义

```
#define NORMAL_DOWN 1
#define ACCELRATE_DOWN 2
#define STOPREFRESH 3
```

规定计时器ID

JudgeUtils.c：定义了函数：

```
void LayerInit()
void UpdateLayers(Tetromino x)
int JudgeBorder(Tetromino x, int FallDirection)
void ClearFullLayer()
int JudgeGameOver(Tetromino x)
```

JudgeUtils.h：

1）声明了JudgeUtils.c中定义的函数；

2）声明了整型常量X_CORNER和Y_CORNER。

ListUtils.c：定义了函数：

```
ListNodePtr CreateList(void)
void freeList(ListNodePtr head)
ListNodePtr InsertNode(ListNodePtr head, int val, char *name)
ListNodePtr DeleteNode(ListNodePtr head)
```

ListUtils.h：

1）声明了ListUtils.c中定义的函数；

2）运用宏定义“#define NAMELENGTH 16”和“#define New(type) ((type) GetBlock(sizeof *((type) NULL)))”

3）定义了类型名为ListNode的结构体并运用typedef将其类型名指定为*ListNodePtr；

SaveUtils.c：定义了函数

void SaveGame()

int RecoverGame()

SaveUtils.h：

声明了SaveUtils.c中定义的函数

3.4.2多文件构成机制

1.各个.c文件所包含的头文件

1) main.c

```
#include ”graphics.h”
#include ”imgui.h”
#include ”DrawUtils.h”
#include <extgraph.h>
#include <windows.h>
#include <random.h>
#include <JudgeUtils.h>
#include <GameUtils.h>
#include <SaveUtils.h>
```

2) DrawUtils.c

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <DrawUtils.h>
#include <ListUtils.h>
#include <JudgeUtils.h>
#include ”graphics.h”
#include ”imgui.h”
#include ”extgraph.h”
#include ”SaveUtils.h”
#include <GameUtils.h>
#include <random.h>
```

3) GameUtils.c

```
#include <graphics.h>
#include <imgui.h>
#include <DrawUtils.h>
#include <extgraph.h>
#include <windows.h>
#include <random.h>
#include <JudgeUtils.h>
#include <GameUtils.h>
#include <ListUtils.h>
```

```
#include <SaveUtils.h>
```

4) JudgeUtils.c

```
#include <stdio.h>
#include <DrawUtils.h>
#include <ListUtils.h>
#include <GameUtils.h>
```

5) SaveUtils.c

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <genlib.h>
#include <GameUtils.h>
```

6) ListUtils.c

```
#include "ListUtils.h"
#include <stdio.h>
#include <stdlib.h>
#include "genlib.h"
#include <string.h>
```

2..h文件中的头文件包含和用#define实现的文件保护

1） DrawUtils.h

利用

```
#ifndef BIGWORK_DRAWUTILS_H
#define BIGWORK_DRAWUTILS_H
```

语句来避免DrawUtils.h被重复包含。类似这样的语句也出现在其他的头文件中，故不赘述。

2） GameUtils.h

包含了DrawUtils.h与ListUtils.h。

3） JudgeUtils.h

包含了DrawUtils.h。

4） ListUtils.h

未包含其他头文件。

5） SaveUtils.h

包含了GameUtils.h。

4.部署与运行

4.1 编译安装运行说明

使用Dev-C++:

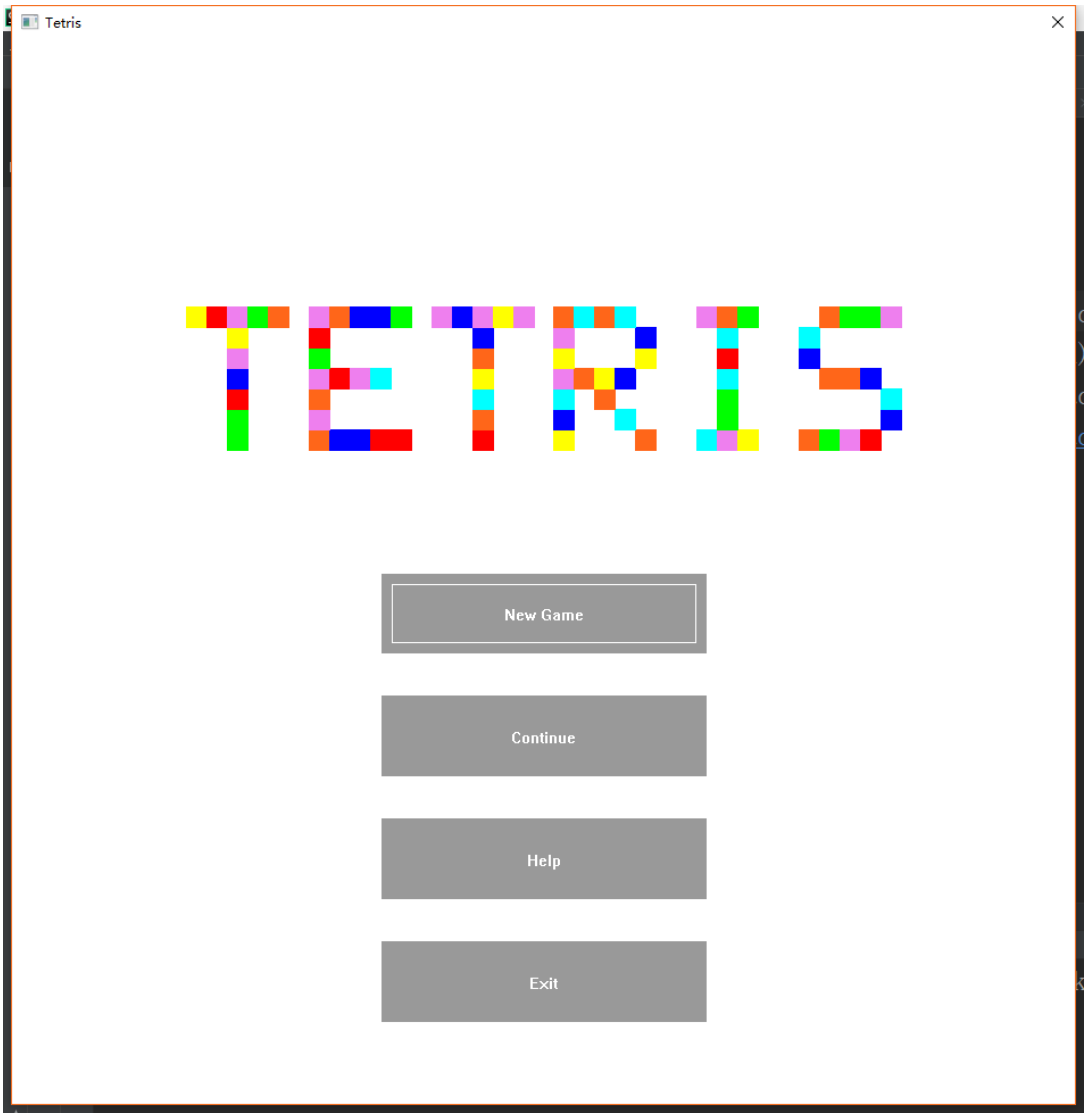
打开Dev-C++，选择打开项目，打开项目所在目录下的Tetris.dev，点击编译运行。

使用Clion:

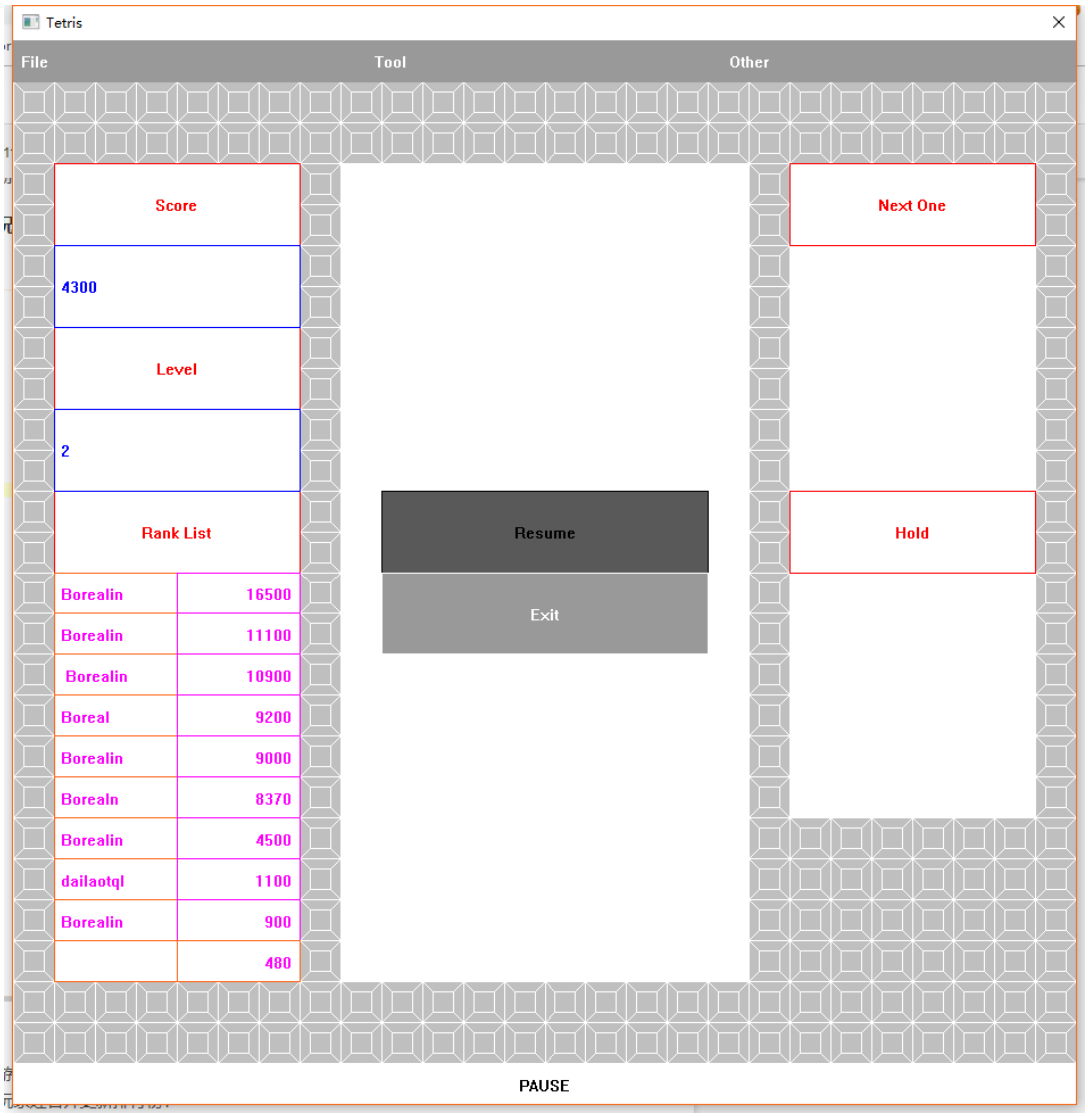
打开Clion，选择打开项目，打开项目所在目录，在左侧文件列表中选择cmakelist.txt，点击load project，成功通过cmakelist加载项目后点击右上角的运行。

4.2典型测试情况

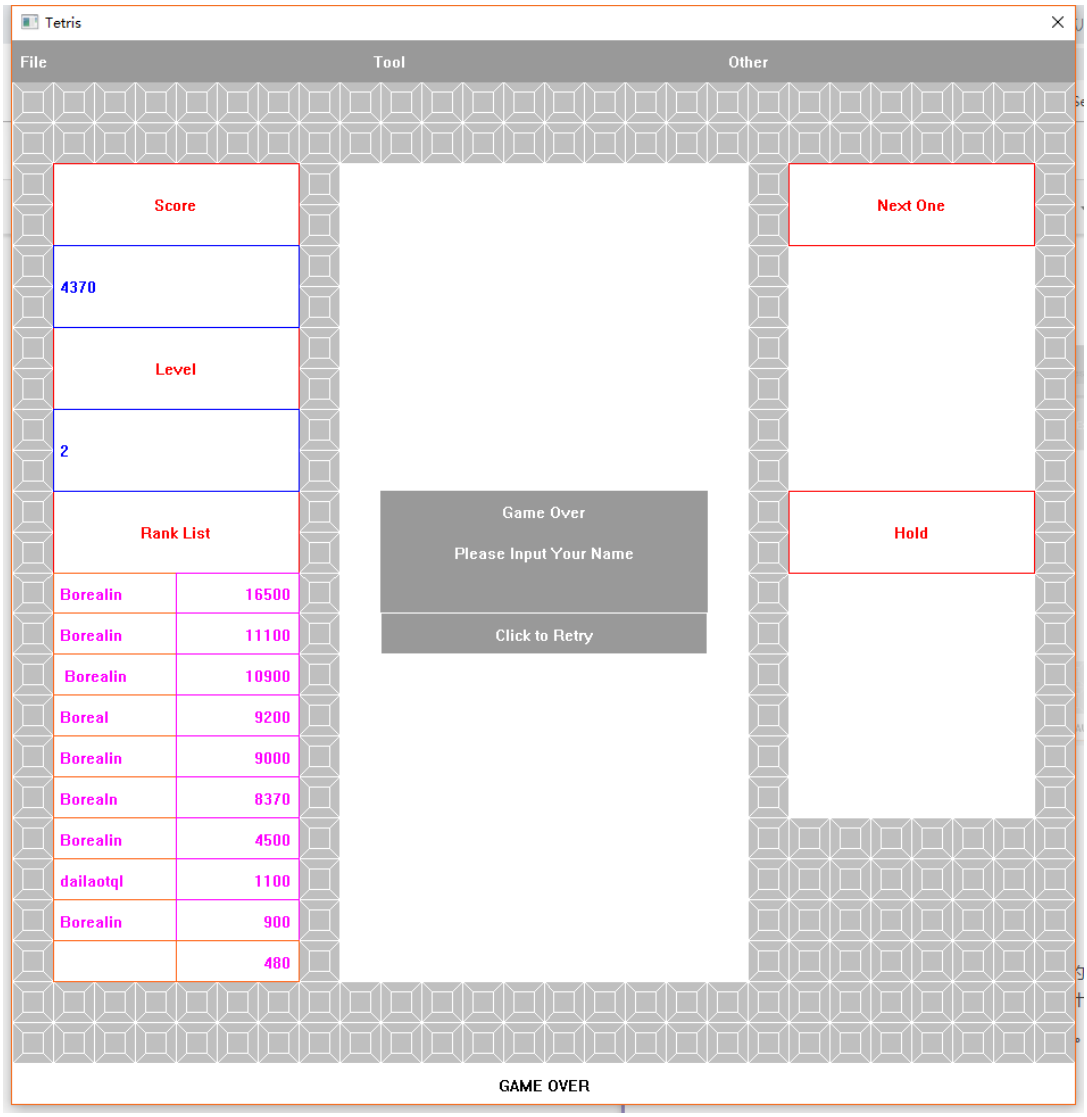
游戏初始界面:



游戏暂停：



游戏结束后输入玩家姓名并更新排行榜：



5.组内分工

5.1 组内分工情况

施楚宁（组长）：对游戏进行数据结构设计，并实现游戏的基本功能；

王逸旦（组员）：撰写代码注释，撰写实验报告，制作ppt。

徐珂（组员）：撰写代码注释，撰写实验报告，尝试设计菜单；

5.2 个人实践过程中遇到的难点及解决方案

组长：

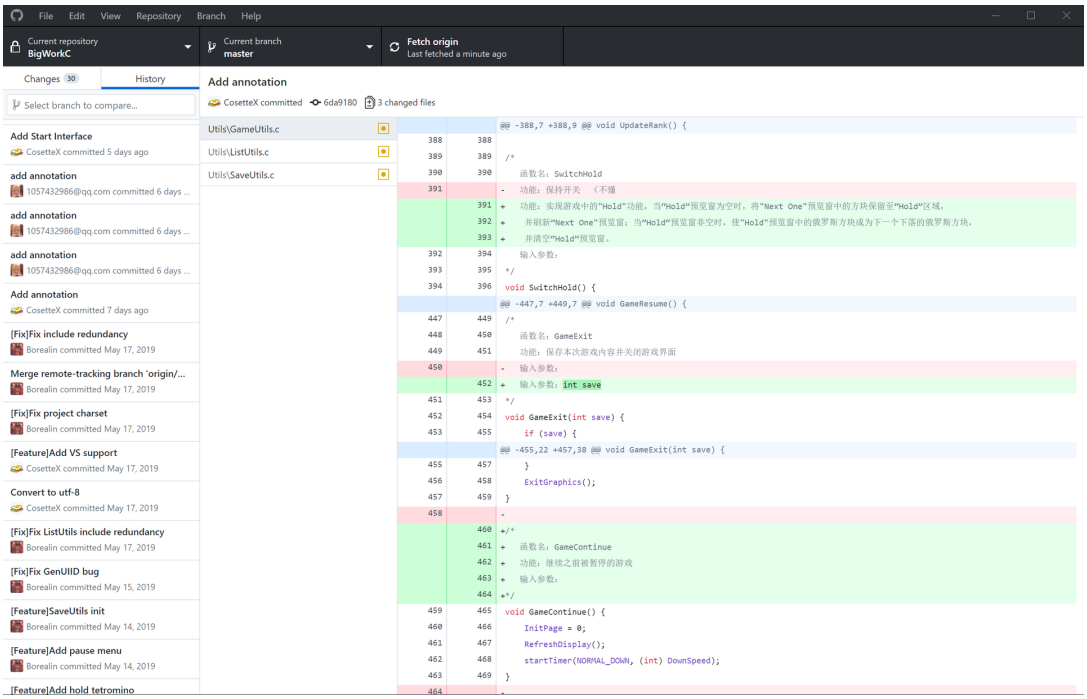
组员：

（1）问题：与组内其他成员共享的代码不适应本地的编译器

解决方案一：使用#define _CRT_SECURE_NO_WARNINGS等宏定义

解决方案二：使用CMake进行编译

6.合作纪要



7.总结

7.1程序亮点或创新之处

程序除了实现俄罗斯方块最基本的功能外，还设有“Hold”功能，并且可以在游戏停止时用“↑”“↓”键调整玩家的等级，增加了游戏的可玩性。在消除满行时设置了闪烁功能，使游戏的视觉体验更加丰富。

7.2应用知识点

1.基于libgraphics库，利用一些常用函数实现了游戏界面的绘制和游戏功能的实现。
2.多处运用数组、指针；
3.运用链表完成了排行榜数据的组织；
4.运用文件相关的知识，实现了游戏的存档和读档功能，并实现了程序总体的多文件组织。

7.3 心得体会

总体描述：

本次小组合作中，大家能够比较及时有效地进行线上或者线下的沟通，基本能保证一周一度的线下讨论，及时地交流进度，讨论对程序需求的设想。较早地实现了游戏的基本功能，并在后期做了一些小幅度的改动。一个小的建议是，为了优化下一次的合作，可以对每一次的讨论做好及时的文字记录，并且可以将一些已经比较成熟的模块早一些记录进实验报告中。

施楚宁：

由于本身已经提前学习过这方面的知识，所以较早的阶段就已经初步实现了游戏的基本功能，之后则是在和两位组员共同讨论的过程中逐步润色、完善游戏的方方面面。在长期的讨论过程中，我们共同提升着编程水平和对游戏的理解。同时我们也能愉快的一起讨论我之前完成的以及手头正在制作的程序。

同时这也是我参与的较为愉快的一次小组合作，大家的参与度和积极度都非常高，都会很积极的参与线上线下的小组讨论。在组员之余，则更像是一种朋友的关系，就很舒服。

徐珂：

这是我第一次通过小组合作制作大程序，给我留下了比较深刻的印象，也让我受益匪浅。在合作的过程中，我能感受到自己与优秀的同学之间的差距。我对代码的理解程度有待加深，编写程序的速度有待提升，编写程序的能力也有待提高。希望自己能在打好课堂基础的前提下更多地学习课外的知识，积累经验。大程制作需要一个比较清晰的前导思路，以便更好地设计程序的文件结构。代码经验比较丰富的同学会预先设置一些量备用，也让我觉得很佩服。这一次的小组合作过程中，组长除了与我们讨论课程作业所要求的程序外，还会分享一些他自己编写的有意思的程序，虽然可能对课程作业没有什么直接的帮助，但是很大地激发了我们两位组员的兴趣，给大家的合作也增添了许多色彩。

王逸旦：

在这次俄罗斯方块的制作中，组长在编写程序之余，还耐心指正我们在编程上的错误，在加深了我们对于编程的理解的同时也并提升了我们对于编译器的应用能力。他也与我们分享了一些学习过程中的经验以及手头上已经完工或正在制作的应用，激发了我们对于学习编程知识的热情。同时，在阅读组长所写的代码的过程中我也积累了对于多文件代码的阅读经验。在这次合作中，我体会到了自身编程能力与课外学习与优秀的同学之间的巨大差距，相信我在之后的学习中会努力加强这方面的学习。

8、参考文献和资料

1.俄罗斯方块_维基百科