#include <conio.h>

#include <iostream.h>

#include <windows.h>

#include <stdlib.h>

#include <ctime>

#include "colorConsole.h" //老师的文件

void begin(); //开始游戏

void frame(); //边框设定

int \* getblocks(); //方块产生

void move(int line); //移动

void drawblocks(int line); //方块显示

void clearsquare(int line); //方块擦出

void turn(int line); //方块旋转

bool isavailable(int line); //判断是否能下落

void remember(int line); //记忆方块位置

void deleteline(int line); //方块满一行消除

bool ifgameover(); //判断是否游戏结束

void end(); //游戏结束

#define up 72

#define down 80

#define left 75

#define right 77

#define esc 27

HANDLE handle;

int a1[4][4]={{1},{1,1,1}}; //七种方块的二维数组

int a2[4][4]={{0,1},{1,1,1}};

int a3[4][4]={{1,1},{0,1,1}};

int a4[4][4]={{0,0,1},{1,1,1}};

int a5[4][4]={{0,1,1},{1,1}};

int a6[4][4]={{1,1,1,1}};

int a7[4][4]={{1,1},{1,1}};

int row=0; //列数

int score=0;

int level=0;

int \* block1=NULL;

int \* block2=NULL;

int \* block3=NULL;

int coordinate[12][18]={0}; //坐标数组，边框12\*18(最后一行，两边边框计算在内)

int judge=0;

int scorex=0;

int temp[4][4]={0};

void main() //主函数

{

int t=1;

handle = initiate();

while(t)

{

t=0;

begin();

sndPlaySound("music.wav",SND\_LOOP|SND\_ASYNC);

frame();

WORD wColors[1];

wColors[0]=FOREGROUND\_GREEN|FOREGROUND\_RED|FOREGROUND\_INTENSITY;

for(int k=1;k<=999999;k++)

{

if(ifgameover()) //判断是否结束

{

textout(handle,34,10,wColors,1,"Game Over");

Sleep(800);

end();

}

else

{

if(k==1)

block2=getblocks();

block3=block2; //block2指向将出现的方块地址

block2=getblocks(); //获取下一个新的方块

block1=block3;

row=52;

clearsquare(16); //擦除next的方块

block1=block2;

drawblocks(15); //在next显示下一块方块图形

row=34;

block1=block3;

for(int i=4;i<=7;i++) //所构建的方块图形最多只有占有两排，所以只用4-7即可对应

{

if(\*(block1+i))

textout(handle,26+i\*2,4,wColors,1,"■"); //方块先露出下面部分

}

Sleep(500-50\*level);

for(int line=4;line<=22;line++) //方块自主下落，方块从第四排开始出现

{

if(isavailable(line)) //检验刚产生的方块是否碰壁，碰到已落方块

{

clearsquare(line); //消除方块先露初的下面分

drawblocks(line); //产生完整的下落方块

move(line);

}

else

{

remember(line); //落定后将这些位置对应的all数组中元素置1

deleteline(line); //消行以及加分

if(line==4)

judge=1;

break;

}

}

}

}

}

}

void begin()

{

int i=1;

WORD wColors[1];

wColors[0]=FOREGROUND\_GREEN|FOREGROUND\_INTENSITY;

WORD wColors1[2];

wColors1[0]=FOREGROUND\_RED|FOREGROUND\_GREEN|FOREGROUND\_INTENSITY;

wColors1[1]=FOREGROUND\_RED|FOREGROUND\_INTENSITY;

textout(handle,18,4,wColors,1," ◢◣ ◢◣");

textout(handle,18,5,wColors,1," ◢ ◎ ◣ ◢ ◎ ◣ ");

textout(handle,18,6,wColors,1,"　　　◢█████████◣");

textout(handle,18,7,wColors,1,"　　██◤＾　　　　　＾◥██");

textout(handle,18,8,wColors,1,"　　██　　　　　　　 　██ ");

textout(handle,18,9,wColors,1,"　◢██　　　　　　　　　██◣");

textout(handle,18,10,wColors,1,"　██◤　●　　　　　●　◥██");

textout(handle,18,11,wColors,1,"　██　◎　　　　　　　◎　██");

textout(handle,18,12,wColors,1,"　◥█◣　　　　Ｔ　　　　◢█◤");

textout(handle,18,13,wColors,1,"　　　██◣　　　　　◢██");

textout(handle,18,14,wColors,1,"　　　　◥███████◤");

textout(handle,18,15,wColors,1," 　　　　 ");

textout(handle,18,16,wColors,1,"　 简单 ◥███████◤ 中等");

textout(handle,18,17,wColors,1,"　请按1　　 █ 请按2");

textout(handle,18,18,wColors,1,"　　　　 █");

textout(handle,18,19,wColors,1,"　　　　 ◢█ █◣");

textout(handle,18,20,wColors,1," 困难 请按 3");

textout(handle,54,22,wColors,1,"MADE BY ");

while(i)

{

textout(handle,30,8,wColors1,2,"俄罗斯方块");

Sleep(800);

textout(handle,30,8,wColors1,2," ");

Sleep(800);

if (\_kbhit()) //输入等级

{

switch(\_getch())

{

case '1':

{

level=1;

i=0; //跳出循环

break;

}

case '2':

{

level=4;

i=0;

break;

}

case '3':

{

level=7;

i=0;

break;

}

}

}

}

system("cls"); //清屏

}

void frame() //边框的设定

{

WORD wColors[1];

wColors[0]=FOREGROUND\_GREEN|FOREGROUND\_BLUE|FOREGROUND\_INTENSITY;

WORD wColors1[1];

wColors1[0]=FOREGROUND\_RED|FOREGROUND\_INTENSITY;

for(int i=0;i<=11;i++)

coordinate[i][17]=1; //底排边框定义为1

for(int j=0;j<=17;j++)

{

coordinate[0][j]=1; //两边边框定义为1

coordinate[11][j]=1;

}

char string[5];

textout(handle,59,5,wColors,1,itoa(level,string,10));

textout(handle,52,5,wColors,1,"level: ");

textout(handle,52,9,wColors,1,"score: 0");

textout(handle,52,13,wColors,1,"next:");

textout(handle,10,6,wColors1,1,"暂停 SPACE");

textout(handle,10,7,wColors1,1,"退出 ESC");

textout(handle,10,8,wColors1,1,"翻转 ↑");

textout(handle,10,9,wColors1,1,"向右 →");

textout(handle,10,10,wColors1,1,"向左 ←");

textout(handle,10,11,wColors1,1,"加速 ↓");

textout(handle,33,2,wColors,1,"来~战个痛");

for(int m=13;m<=24;m++)

{

textout(handle,2\*m,3,wColors,1,"═"); //上边框

}

for(int n=4;n<=21;n++)

{

textout(handle,26,n,wColors,1,"‖"); //左边框

}

for(int k=4;k<=21;k++)

{

textout(handle,48,k,wColors,1,"‖"); //右边框

}

for(int l=13;l<=23;l++)

{

textout(handle,2\*l,21,wColors,1,"═"); //下边框

}

textout(handle,26,3,wColors,1,"◤");

textout(handle,48,3,wColors,1,"◥");

textout(handle,26,21,wColors,1,"◣");

textout(handle,48,21,wColors,1,"◢");

}

int \* getblocks() //随机方块生成

{

int \* m=NULL;

srand(time(NULL));

int n=rand()%7;

switch(n)

{

case 0:

m=&a1[0][0];break;

case 1:

m=&a2[0][0];break;

case 2:

m=&a3[0][0];break;

case 3:

m=&a4[0][0];break;

case 4:

m=&a5[0][0];break;

case 5:

m=&a6[0][0];break;

case 6:

m=&a7[0][0];break;

}

return m;

}

void drawblocks(int line) //出现方块

{

WORD wColors[1];

wColors[0]=FOREGROUND\_GREEN|FOREGROUND\_RED|FOREGROUND\_INTENSITY;

for(int j=0;j<=15;j++)

{

int temp;

temp=j/4;

if(\*(block1+j))

textout(handle,row+j\*2-temp\*8,line+temp,wColors,1,"■");

}

}

void clearsquare(int line) //方块消失

{

WORD wColors[1];

wColors[0]=FOREGROUND\_BLUE|FOREGROUND\_INTENSITY;

if(line==4) //针对消除刚产生的下排

{

textout(handle,34,4,wColors,1," ");

textout(handle,36,4,wColors,1," ");

textout(handle,38,4,wColors,1," ");

textout(handle,40,4,wColors,1," ");

}

else

{

for(int m=0;m<=15;m++)

{

int temp;

temp=m/4; //得0-3对应方块数组1-4行

if(\*(block1+m))

textout(handle,row+m\*2-temp\*8,line-1+temp,wColors,1," ");

}

}

}

void move(int line) //方块的左右移动，加速下落，翻转等

{

int mid=0,speed=100-10\*level;

while(mid<speed)

{

if (\_kbhit())

{

switch(\_getch())

{

case 72: //翻转

{

turn(line);

break;

}

case 75: //左移

{

row=row-2; //纵坐标减2

if(isavailable(line)) //判断是否能移动

{

row=row+2;

clearsquare(line+1); //消除原来图案，line+1是避免line=4程序出错

row=row-2;

drawblocks(line); //出现新图案

}

else

row=row+2; //若不能移动则纵坐标不变

break;

}

case 77: //右移

{

row=row+2;

if(isavailable(line))

{

row=row-2;

clearsquare(line+1);

row=row+2;

drawblocks(line);

}

else

row=row-2;

break;

}

case 80: //加速下落，即直接跳除循环

{

mid=speed;

break;

}

case 27: //终止游戏

{

end();

break;

}

case 32: //暂停

{

int flag=1;

while(flag)

{

if (\_kbhit())

{

if(\_getch()==32)

flag=0;

break;

}

else

Sleep(10);

}

}

default:

break;

}

}

Sleep(8); //使方块延迟

mid++;

}

}

void turn(int line)

{

clearsquare(line+1); //消除原来的图案

int b[4][4]={0}; //保存旋转前的方块

int num=0,l=0;

for(int m=0;m<=3;m++)

{

for(int n=0;n<=3;n++)

{

b[m][n]=\*(block1+m\*4+n); //把b[4][4]全赋值为当前图形数组

temp[m][n]=0;

}

}

for(int i=3;i>=0;i--) //按行从下向上扫描

{

for(int j=0;j<4;j++) //按列从左向右扫描

{

if(b[i][j]) //如果为有效点，则进行90度旋转

{

temp[j][l]=b[i][j];

num=1;

}

}

if(num)

{

l++;

num=0;

}

}

block1=&temp[0][0];

if(isavailable(line))

drawblocks(line);

else

{

for(int p=0;p<=3;p++)

{

for(int q=0;q<=3;q++)

temp[p][q]=b[p][q];

}

block1=&temp[0][0];

drawblocks(line);

}

}

bool isavailable(int line) //检验，即看方块即将存在位置是否已经有1

{

int x,y;

for(int m=0;m<=15;m++)

{

int temp;

temp=m/4;

x=row/2-13-4\*temp+m; //边框左边已有13个位置

y=line-4+temp; //上面已有4个位置

if(\*(block1+m)&&coordinate[x][y]) //相与为1则返回0，否则跳出并循环继续

return 0;

}

}

void remember(int line) //记忆

{

int x,y;

for(int m=0;m<=15;m++)

{

int temp;

temp=m/4;

x=row/2-13-temp\*4+m;

y=line-4+temp;

if(\*(block1+m)) //如果当前位置为1，则返回原位置，并设置为1

coordinate[x][y-1]=1;

}

}

void deleteline(int l) //消行

{

WORD wColors[1];

wColors[0]=FOREGROUND\_GREEN|FOREGROUND\_RED|FOREGROUND\_INTENSITY;

int snum=0,b=0;

for(int m=0;m<=16;m++) //从上向下消去方块

{

if(coordinate[1][m]==1&&coordinate[2][m]==1&&coordinate[3][m]==1&&coordinate[4][m]==1&&coordinate[5][m]==1&&coordinate[6][m]==1&&coordinate[7][m]==1&&coordinate[8][m]==1&&coordinate[9][m]==1&&coordinate[10][m]==1)

{

textout(handle,28,m+4,wColors,1,"﹌﹌﹌﹌good﹌﹌﹌﹌");

Sleep(750);

for(int n=1;n<=m;n++)

{

for(int j=1;j<=10;j++)

coordinate[j][m-n+1]=coordinate[j][m-n];

}

snum++;

}

}

for(int n=1;n<=10;n++)

{

for(int d=0;d<=16;d++)

{

int x,y;

x=n\*2+26;

y=d+4;

textout(handle,x,y,wColors,1," ");

if(coordinate[n][d])

{

textout(handle,x,y,wColors,1,"■");

}

}

}

score=score+(snum\*(snum+1)/2);

if((score-scorex)>=30) //每得到30分自动加速

{

level++;

scorex=score;

}

char string[5];

textout(handle,59,9,wColors,1,itoa(score,string,10));

textout(handle,59,5,wColors,1,itoa(level,string,10));

}

bool ifgameover() //终止游戏

{

if(judge==1)

{

return 1;

}

else

return 0;

}

void end() //退出

{

WORD wColors[1];

wColors[0]=FOREGROUND\_GREEN|FOREGROUND\_INTENSITY;

textout(handle,28,22,wColors,1,"Press any key to exit");

while(1)

{

if (\_kbhit())

{

exit(EXIT\_SUCCESS);

}

}

}