#include <stdio.h> 搜索  
#include <dos.h>   
#include <conio.h>   
#include <graphics.h>   
#include <stdlib.h>   
#ifdef \_\_cplusplus   
#define \_\_CPPARGS ...   
#else   
#define \_\_CPPARGS   
#endif   
#define MINBOXSIZE 15 /\* 最小方块的尺寸 \*/   
#define BGCOLOR 7 /\* 背景着色 \*/   
#define GX 200   
#define GY 10   
#define SJNUM 10000 /\* 每当玩家打到一万分等级加一级\*/   
/\* 按键码\*/   
#define VK\_LEFT 0x4b00   
#define VK\_RIGHT 0x4d00   
#define VK\_DOWN 0x5000   
#define VK\_UP 0x4800   
#define VK\_HOME 0x4700   
#define VK\_END 0x4f00   
#define VK\_SPACE 0x3920   
#define VK\_ESC 0x011b   
#define VK\_ENTER 0x1c0d   
/\* 定义[俄罗斯方块](https://www.baidu.com/s?wd=%E4%BF%84%E7%BD%97%E6%96%AF%E6%96%B9%E5%9D%97&tn=67012150_cpr&fenlei=mv6quAkxTZn0IZRqIHcvrjTdrjb0T1dBPHnvm1Rzmv7bm1nYnWN90ZwV5fKWUMw85HmLnjDznHRsgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1csn16sPWbd)的方向（我定义他为4种）\*/   
#define F\_DONG 0   
#define F\_NAN 1   
#define F\_XI 2   
#define F\_BEI 3   
#define NEXTCOL 20 /\* 要出的下一个方块的纵坐标\*/   
#define NEXTROW 12 /\* 要出的下一个方块的横从标\*/   
#define MAXROW 14 /\* 游戏屏幕大小\*/   
#define MAXCOL 20   
#define SCCOL 100 /\*游戏屏幕大显示器上的相对位置\*/   
#define SCROW 60   
  
int gril[22][16]; /\* 游戏屏幕坐标\*/   
int col=1,row=7; /\* 当前方块的横纵坐标\*/   
int boxfx=0,boxgs=0; /\* 当前寺块的形壮和方向\*/   
int nextboxfx=0,nextboxgs=0,maxcol=22;/\*下一个方块的形壮和方向\*/   
int minboxcolor=6,nextminboxcolor=6;   
int num=0; /\*游戏分\*/   
int dj=0,gamedj[10]={18,16,14,12,10,8,6,4,2,1};/\* 游戏等级\*/   
/\* 以下我用了一个3维数组来纪录方块的最初形状和方向\*/   
int boxstr[7][4][16]={{   
{1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0},   
{0,1,0,0,1,1,0,0,1,0,0,0,0,0,0,0},   
{1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0},   
{0,1,0,0,1,1,0,0,1,0,0,0,0,0,0,0}},   
{   
{0,1,1,0,1,1,0,0,0,0,0,0,0,0,0,0},   
{1,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0},   
{0,1,1,0,1,1,0,0,0,0,0,0,0,0,0,0},   
{1,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0}},   
{   
{1,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0},   
{1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0},   
{1,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0},   
{0,0,1,0,1,1,1,0,0,0,0,0,0,0,0,0}},   
{   
{1,1,0,0,1,0,0,0,1,0,0,0,0,0,0,0},   
{1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0},   
{0,1,0,0,0,1,0,0,1,1,0,0,0,0,0,0},   
{1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0}},   
{   
{0,1,0,0,0,1,0,0,0,1,0,0,0,1,0,0},   
{0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0},   
{0,1,0,0,0,1,0,0,0,1,0,0,0,1,0,0},   
{0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0}},   
{   
{1,1,0,0,1,1,0,0,0,0,0,0.0,0,0,0},   
{1,1,0,0,1,1,0,0,0,0,0,0.0,0,0,0},   
{1,1,0,0,1,1,0,0,0,0,0,0.0,0,0,0},   
{1,1,0,0,1,1,0,0,0,0,0,0.0,0,0,0}},   
{   
{0,0,0,0,1,1,1,0,0,1,0,0,0,0,0,0},   
{1,0,0,0,1,1,0,0,1,0,0,0,0,0,0,0},   
{0,1,0,0,1,1,1,0,0,0,0,0.0,0,0,0},   
{0,1,0,0,1,1,0,0,0,1,0,0,0,0,0,0}}   
};   
/\* 随机得到当前方块和下一个方块的形状和方向\*/   
void boxrad(){   
minboxcolor=nextminboxcolor;   
boxgs=nextboxgs;   
boxfx=nextboxfx;   
nextminboxcolor=random(14)+1;   
if(nextminboxcolor==4||nextminboxcolor==7||nextminboxcolor==8)   
nextminboxcolor=9;   
nextboxfx=F\_DONG;   
nextboxgs=random(7);   
}   
/\*初始化图形模试\*/   
void init(int gdrive,int gmode){   
int errorcode;   
initgraph(&gdrive,&gmode,"e:\\tc");   
errorcode=graphresult();   
if(errorcode!=grOk){   
printf("error of: %s",grapherrormsg(errorcode));   
exit(1);   
}   
}   
/\* 在图形模式下的清屏 \*/   
void cls()   
{   
setfillstyle(SOLID\_FILL,0);   
setcolor(0);   
bar(0,0,640,480);   
}   
/\*在图形模式下的高级清屏\*/   
void clscr(int a,int b,int c,int d,int color){   
setfillstyle(SOLID\_FILL,color);   
setcolor(color);   
bar(a,b,c,d);   
}   
/\*最小方块的绘制\*/   
void minbox(int asc,int bsc,int color,int bdcolor){   
int a=0,b=0;   
a=SCCOL+asc;   
b=SCROW+bsc;   
clscr(a+1,b+1,a-1+MINBOXSIZE,b-1+MINBOXSIZE,color);   
if(color!=BGCOLOR){   
setcolor(bdcolor);   
line(a+1,b+1,a-1+MINBOXSIZE,b+1);   
line(a+1,b+1,a+1,b-1+MINBOXSIZE);   
line(a-1+MINBOXSIZE,b+1,a-1+MINBOXSIZE,b-1+MINBOXSIZE);   
line(a+1,b-1+MINBOXSIZE,a-1+MINBOXSIZE,b-1+MINBOXSIZE);   
}   
}   
/\*游戏中出现的文字\*/   
void txt(int a,int b,char \*txt,int font,int color){   
setcolor(color);   
settextstyle(0,0,font);   
outtextxy(a,b,txt);   
}   
/\*windows 绘制\*/   
void win(int a,int b,int c,int d,int bgcolor,int bordercolor){   
clscr(a,b,c,d,bgcolor);   
setcolor(bordercolor);   
line(a,b,c,b);   
line(a,b,a,d);   
line(a,d,c,d);   
line(c,b,c,d);   
}   
/\* 当前方块的绘制\*/   
void funbox(int a,int b,int color,int bdcolor){   
int i,j;   
int boxz[4][4];   
for(i=0;i<16;i++)   
boxz[i/4][i%4]=boxstr[boxgs][boxfx][i];  
for(i=0;i<4;i++)   
for(j=0;j<4;j++)   
if(boxz[i][j]==1)  
minbox((j+row+a)\*MINBOXSIZE,(i+col+b)\*MINBOXSIZE,color,bdcolor);   
}   
/\*下一个方块的绘制\*/   
void nextfunbox(int a,int b,int color,int bdcolor){   
int i,j;   
int boxz[4][4];   
for(i=0;i<16;i++)   
boxz[i/4][i%4]=boxstr[nextboxgs][nextboxfx][i];  
for(i=0;i<4;i++)   
for(j=0;j<4;j++)   
if(boxz[i][j]==1)  
minbox((j+a)\*MINBOXSIZE,(i+b)\*MINBOXSIZE,color,bdcolor);   
}   
/\*时间中断定义\*/   
#define TIMER 0x1c   
int TimerCounter=0;   
void interrupt ( \*oldhandler)(\_\_CPPARGS);   
void interrupt newhandler(\_\_CPPARGS){   
TimerCounter++;   
oldhandler();   
}   
void SetTimer(void interrupt (\*IntProc)(\_\_CPPARGS)){   
oldhandler=getvect(TIMER);   
disable();   
setvect(TIMER,IntProc);   
enable();   
}   
/\*由于游戏的规则，消掉都有最小方块的一行\*/   
void delcol(int a){   
int i,j;   
for(i=a;i>1;i--)   
for(j=1;j<15;j++){   
minbox(j\*MINBOXSIZE,i\*MINBOXSIZE,BGCOLOR,BGCOLOR);   
gril[i][j]=gril[i-1][j];  
if(gril[i][j]==1)  
minbox(j\*MINBOXSIZE,i\*MINBOXSIZE,minboxcolor,0);   
}   
}   
/\*消掉所有都有最小方块的行\*/   
void delete(){   
int i,j,zero,delgx=0;   
char \*nm="00000";   
for(i=1;i<21;i++){   
zero=0;   
for(j=1;j<15;j++)   
if(gril[j]==0)   
zero=1;   
if(zero==0){   
delcol(i);   
delgx++;   
}   
}   
num=num+delgx\*delgx\*10;   
dj=num/10000;   
sprintf(nm,"%d",num);   
clscr(456,173,500,200,4);   
txt(456,173,"Number:",1,15);   
txt(456,193,nm,1,15);   
}   
/\*时间中断结束\*/   
void KillTimer(){   
disable();   
setvect(TIMER,oldhandler);   
enable();   
}   
/\* 测试当前方块是否可以向下落\*/   
int downok(){   
int i,j,k=1,a[4][4];   
for(i=0;i<16;i++)   
a[i/4][i%4]=boxstr[boxgs][boxfx][i];  
for(i=0;i<4;i++)   
for(j=0;j<4;j++)   
if(a[j] && gril[col+i+1][row+j])   
k=0;   
return(k);   
}   
/\* 测试当前方块是否可以向左行\*/   
int leftok(){   
int i,j,k=1,a[4][4];   
for(i=0;i<16;i++)   
a[i/4][i%4]=boxstr[boxgs][boxfx][i];  
for(i=0;i<4;i++)   
for(j=0;j<4;j++)   
if(a[j] && gril[col+i][row+j-1])   
k=0;   
return(k);   
}   
/\* 测试当前方块是否可以向右行\*/   
int rightok(){   
int i,j,k=1,a[4][4];   
for(i=0;i<16;i++)   
a[i/4][i%4]=boxstr[boxgs][boxfx][i];  
for(i=0;i<4;i++)   
for(j=0;j<4;j++)   
if(a[j] && gril[col+i][row+j+1])   
k=0;   
return(k);   
}   
/\* 测试当前方块是否可以变形\*/   
int upok(){   
int i,j,k=1,a[4][4];   
for(i=0;i<4;i++)   
for(i=0;i<16;i++)   
a[i/4][i%4]=boxstr[boxgs][boxfx+1][i];  
for(i=3;i>=0;i--)   
for(j=3;j>=0;j--)   
if(a[j] && gril[col+i][row+j])   
k=0;   
return(k);   
}   
/\*当前方块落下之后，给屏幕坐标作标记\*/   
void setgril(){   
int i,j,a[4][4];   
funbox(0,0,minboxcolor,0);   
for(i=0;i<16;i++)   
a[i/4][i%4]=boxstr[boxgs][boxfx][i];  
for(i=0;i<4;i++)   
for(j=0;j<4;j++)   
if(a[j])   
gril[col+i][row+j]=1;   
col=1;row=7;   
}   
/\*游戏结束\*/   
void gameover(){   
int i,j;   
for(i=20;i>0;i--)   
for(j=1;j<15;j++)   
minbox(j\*MINBOXSIZE,i\*MINBOXSIZE,2,0);   
txt(103,203,"Game Over",3,10);   
}   
/\*按键的设置\*/   
void call\_key(int keyx){   
switch(keyx){   
case VK\_DOWN: { /\*下方向键，横坐标加一。\*/   
if(downok()){   
col++;   
funbox(0,0,minboxcolor,0);}   
else{   
funbox(0,0,minboxcolor,0);   
setgril();   
nextfunbox(NEXTCOL,NEXTROW,4,4);   
boxrad();   
nextfunbox(NEXTCOL,NEXTROW,nextminboxcolor,0);   
delete();   
}   
break;   
}   
case VK\_UP: { /\*上方向键，方向形状旋转90度\*/   
if(upok())   
boxfx++;   
if(boxfx>3)   
boxfx=0;   
funbox(0,0,minboxcolor,0);   
break;   
}   
case VK\_LEFT:{ /\*左方向键，纵坐标减一\*/   
if(leftok())   
row--;   
funbox(0,0,minboxcolor,0);   
break;   
}   
case VK\_RIGHT:{ /\*右方向键，纵坐标加一\*/   
if(rightok())   
row++;   
funbox(0,0,minboxcolor,0);   
break;   
}   
case VK\_SPACE: /\*空格键，直接落到最后可以落到的们置\*/   
while(downok())   
col++;   
funbox(0,0,minboxcolor,0);   
setgril();   
nextfunbox(NEXTCOL,NEXTROW,4,4);   
boxrad();   
nextfunbox(NEXTCOL,NEXTROW,nextminboxcolor,0);   
delete();   
break;   
default:   
{   
txt(423,53,"worng key!",1,4);   
txt(428,80,"Plese Enter Anly Key AG!",1,4);   
getch();   
clscr(420,50,622,97,BGCOLOR);   
}   
}   
}   
/\*时间中断开始\*/   
void timezd(void){   
int key;   
SetTimer(newhandler);   
boxrad();   
nextfunbox(NEXTCOL,NEXTROW,nextminboxcolor,0);   
for(;;){   
if(bioskey(1)){   
key=bioskey(0);   
funbox(0,0,BGCOLOR,BGCOLOR);   
if(key==VK\_ESC)   
break;   
call\_key(key);   
}   
if(TimerCounter>gamedj[dj]){   
TimerCounter=0;   
if(downok()){   
funbox(0,0,BGCOLOR,BGCOLOR);   
col++;   
funbox(0,0,minboxcolor,0);   
}   
else {   
if(col==1){   
gameover();   
getch();   
break;   
}   
setgril();   
delete();   
funbox(0,0,minboxcolor,0);   
col=1;row=7;   
funbox(0,0,BGCOLOR,BGCOLOR);   
nextfunbox(NEXTCOL,NEXTROW,4,4);   
boxrad();   
nextfunbox(NEXTCOL,NEXTROW,nextminboxcolor,0);   
}   
}   
}   
}   
/\*主程序开始\*/   
void main(void){   
int i,j;   
char \*nm="00000";   
init(VGA,VGAHI);   
cls();   
/\*屏幕坐标初始化\*/   
for(i=0;i<=MAXCOL+1;i++)   
for(j=0;j<=MAXROW+1;j++)   
gril[i][j]=0;  
for(i=0;i<=MAXCOL+1;i++) {   
gril[i][0]=1;  
gril[i][15]=1;  
}   
for(j=1;j<=MAXROW;j++){   
gril[0][j]=1;   
gril[21][j]=1;   
}   
clscr(0,0,640,480,15);   
win(1,1,639,479,4,15);   
win(SCCOL+MINBOXSIZE-2,SCROW+MINBOXSIZE-2,SCCOL+15\*MINBOXSIZE+2,SCROW+21\*MINBOXSIZE+2,BGCOLOR,0);   
nextboxgs=random(8);   
nextboxfx=random(4);   
sprintf(nm,"%d",num);   
txt(456,173,"Number:",1,15);   
txt(456,193,nm,1,15);   
txt(456,243,"Next Box:",1,15);   
timezd();   
KillTimer();   
closegraph();   
getch();  
}