```
#include <graphics.h>
#include <stdlib.h>
#include <dos.h>
#define LEFTPRESS 0xff01
#define LEFTCLICK 0xff10
#define LEFTDRAG 0xff19
#define MOUSEMOVE 0xff08
struct
{
   int num;/*格子当前处于什么状态,1 有雷，0 已经显示过数字或者空白格子*/
   int roundnum;/*统计格子周围有多少雷*/
   int flag;/*右键按下显示红旗的标志,0 没有红旗标志,1 有红旗标志*/
}Mine[10][10];
int gameAGAIN=0;/*是否重来的变量*/
int gamePLAY=0;/*是否是第一次玩游戏的标志*/
int mineNUM;/*统计处理过的格子数*/
char randmineNUM[3];/*显示数字的字符串*/
int Keystate;
int MouseExist;
int MouseButton;
int MouseX;
int MouseY;
void Init(void);/*图形驱动*/
void MouseOn(void);/*鼠标光标显示*/
void MouseOff(void);/*鼠标光标隐藏*/
void MouseSetXY(int,int);/*设置当前位置*/
int  LeftPress(void);/*左键按下*/
int  RightPress(void);/*鼠标右键按下*/
void MouseGetXY(void);/*得到当前位置*/
void Control(void);/*游戏开始,重新,关闭*/
void GameBegain(void);/*游戏开始画面*/
void DrawSmile(void);/*画笑脸*/
void DrawRedflag(int,int);/*显示红旗*/
void DrawEmpty(int,int,int,int);/*两种空格子的显示*/
void GameOver(void);/*游戏结束*/
void GameWin(void);/*显示胜利*/
int  MineStatistics(int,int);/*统计每个格子周围的雷数*/
int  ShowWhite(int,int);/*显示无雷区的空白部分*/
void GamePlay(void);/*游戏过程*/
void Close(void);/*图形关闭*/
void main(void)
{
   Init();
   Control();
```

```
    Close();
}
void Init(void)/*图形开始*/
{
    int gd=DETECT,gm;
    initgraph(&gd,&gm,"c:\\tc");
}
void Close(void)/*图形关闭*/
{
    closegraph();
}
void MouseOn(void)/*鼠标光标显示*/
{
    _AX=0x01;
    geninterrupt(0x33);
}
void MouseOff(void)/*鼠标光标隐藏*/
{
    _AX=0x02;
    geninterrupt(0x33);
}
void MouseSetXY(int x,int y)/*设置当前位置*/
{
    _CX=x;
    _DX=y;
    _AX=0x04;
    geninterrupt(0x33);
}
int LeftPress(void)/*鼠标左键按下*/
{
    _AX=0x03;
    geninterrupt(0x33);
    return(_BX&1);
}
int RightPress(void)/*鼠标右键按下*/
{
    _AX=0x03;
    geninterrupt(0x33);
    return(_BX&2);
}
void MouseGetXY(void)/*得到当前位置*/
{
    _AX=0x03;
    geninterrupt(0x33);
```

```
    MouseX=_CX;
    MouseY=_DX;
}
void Control(void)/*游戏开始,重新,关闭*/
{
    int gameFLAG=1;/*游戏失败后判断是否重新开始的标志*/
    while(1)
    {
       if(gameFLAG)/*游戏失败后没判断出重新开始或者退出游戏的话就继续判断*/
       {
      GameBegain(); /*游戏初始画面*/
      GamePlay();/*具体游戏*/
      if(gameAGAIN==1)/*游戏中重新开始*/
      {
         gameAGAIN=0;
         continue;
      }
       }
    MouseOn();
    gameFLAG=0;
    if(LeftPress())/*判断是否重新开始*/
    {
       MouseGetXY();
       if(MouseX>280&&MouseX<300&&MouseY>65&&MouseY<85)
       {
      gameFLAG=1;
      continue;
       }
    }
    if(kbhit())/*判断是否按键退出*/
       break;
    }
    MouseOff();
}
void DrawSmile(void)/*画笑脸*/
{
    setfillstyle(SOLID_FILL,YELLOW);
    fillellipse(290,75,10,10);
    setcolor(YELLOW);
    setfillstyle(SOLID_FILL,BLACK);/*眼睛*/
    fillellipse(285,75,2,2);
    fillellipse(295,75,2,2);
    setcolor(BLACK);/*嘴巴*/
    bar(287,80,293,81);
```

```
}
void DrawRedflag(int i,int j)/*显示红旗*/
{
   setcolor(7);
   setfillstyle(SOLID_FILL,RED);
   bar(198+j*20,95+i*20,198+j*20+5,95+i*20+5);
   setcolor(BLACK);
   line(198+j*20,95+i*20,198+j*20,95+i*20+10);
}
void DrawEmpty(int i,int j,int mode,int color)/*两种空格子的显示*/
{
   setcolor(color);
   setfillstyle(SOLID_FILL,color);
   if(mode==0)/*没有单击过的大格子*/
      bar(200+j*20-8,100+i*20-8,200+j*20+8,100+i*20+8);
   else
      if(mode==1)/*单击过后显示空白的小格子*/
     bar(200+j*20-7,100+i*20-7,200+j*20+7,100+i*20+7);
}
void GameBegain(void)/*游戏开始画面*/
{
   int i,j;
   cleardevice();
   if(gamePLAY!=1)
   {
      MouseSetXY(290,70); /*鼠标一开始的位置,并作为它的初始坐标*/
      MouseX=290;
      MouseY=70;
   }
   gamePLAY=1;/*下次按重新开始的话鼠标不重新初始化*/
   mineNUM=0;
   setfillstyle(SOLID_FILL,7);
   bar(190,60,390,290);
   for(i=0;i<10;i++)/*画格子*/
      for(j=0;j<10;j++)
     DrawEmpty(i,j,0,8);
   setcolor(7);
   DrawSmile();/*画脸*/
   randomize();
   for(i=0;i<10;i++)
   /*100 个格子随机赋值有没有地雷*/
      for(j=0;j<10;j++)
      {
     Mine[i][j].num=random(8);/*如果随机数的结果是 1 表示这个格子有地雷*/
```

```
      if(Mine[i][j].num==1)
         mineNUM++;/*现有雷数加 1*/
      else
         Mine[i][j].num=2;
      Mine[i][j].flag=0;/*表示没红旗标志*/
       }
   sprintf(randmineNUM,"%d",mineNUM); /*显示这次总共有多少雷数*/
   setcolor(1);
   settextstyle(0,0,2);
   outtextxy(210,70,randmineNUM);
   mineNUM=100-mineNUM;/*变量取空白格数量*/
   MouseOn();
}
void GameOver(void)/*游戏结束画面*/
{
   int i,j;
   setcolor(0);
   for(i=0;i<10;i++)
      for(j=0;j<10;j++)
      if(Mine[i][j].num==1)/*显示所有的地雷*/
      {
         DrawEmpty(i,j,0,RED);
         setfillstyle(SOLID_FILL,BLACK);
         fillellipse(200+j*20,100+i*20,7,7);
      }
}
void GameWin(void)/*显示胜利*/
{
   setcolor(11);
   settextstyle(0,0,2);
   outtextxy(230,30,"YOU WIN!");
}
int MineStatistics(int i,int j)/*统计每个格子周围的雷数*/
{
   int nNUM=0;
   if(i==0&&j==0)/*左上角格子的统计*/
   {
      if(Mine[0][1].num==1)
      nNUM++;
      if(Mine[1][0].num==1)
      nNUM++;
      if(Mine[1][1].num==1)
      nNUM++;
   }
```

```
else
   if(i==0&&j==9)/*右上角格子的统计*/
    {
  if(Mine[0][8].num==1)
     nNUM++;
  if(Mine[1][9].num==1)
     nNUM++;
  if(Mine[1][8].num==1)
     nNUM++;
   }
  else
  if(i==9&&j==0)/*左下角格子的统计*/
  {
     if(Mine[8][0].num==1)
        nNUM++;
     if(Mine[9][1].num==1)
        nNUM++;
     if(Mine[8][1].num==1)
        nNUM++;
  }
 else
     if(i==9&&j==9)/*右下角格子的统计*/
     {
        if(Mine[9][8].num==1)
       nNUM++;
        if(Mine[8][9].num==1)
       nNUM++;
        if(Mine[8][8].num==1)
       nNUM++;
     }
     else if(j==0)/*左边第一列格子的统计*/
     {
        if(Mine[i][j+1].num==1)
       nNUM++;
        if(Mine[i+1][j].num==1)
       nNUM++;
        if(Mine[i-1][j].num==1)
       nNUM++;
        if(Mine[i-1][j+1].num==1)
       nNUM++;
        if(Mine[i+1][j+1].num==1)
       nNUM++;
     }
     else if(j==9)/*右边第一列格子的统计*/
```

```
{
   if(Mine[i][j-1].num==1)
  nNUM++;
   if(Mine[i+1][j].num==1)
  nNUM++;
   if(Mine[i-1][j].num==1)
  nNUM++;
   if(Mine[i-1][j-1].num==1)
  nNUM++;
   if(Mine[i+1][j-1].num==1)
  nNUM++;
}
else if(i==0)/*第一行格子的统计*/
{
   if(Mine[i+1][j].num==1)
  nNUM++;
   if(Mine[i][j-1].num==1)
  nNUM++;
   if(Mine[i][j+1].num==1)
  nNUM++;
   if(Mine[i+1][j-1].num==1)
  nNUM++;
   if(Mine[i+1][j+1].num==1)
  nNUM++;
 }
 else if(i==9)/*最后一行格子的统计*/
 {
   if(Mine[i-1][j].num==1)
  nNUM++;
   if(Mine[i][j-1].num==1)
  nNUM++;
   if(Mine[i][j+1].num==1)
  nNUM++;
   if(Mine[i-1][j-1].num==1)
  nNUM++;
   if(Mine[i-1][j+1].num==1)
  nNUM++;
}
else/*普通格子的统计*/
{
   if(Mine[i-1][j].num==1)
  nNUM++;
   if(Mine[i-1][j+1].num==1)
  nNUM++;
```

```
            if(Mine[i][j+1].num==1)
           nNUM++;
            if(Mine[i+1][j+1].num==1)
           nNUM++;
            if(Mine[i+1][j].num==1)
           nNUM++;
            if(Mine[i+1][j-1].num==1)
           nNUM++;
            if(Mine[i][j-1].num==1)g
           nNUM++;
            if(Mine[i-1][j-1].num==1)
           nNUM++;
          }
   return(nNUM);/*把格子周围一共有多少雷数的统计结果返回*/
}
int ShowWhite(int i,int j)/*显示无雷区的空白部分*/
{
   if(Mine[i][j].flag==1||Mine[i][j].num==0)/*如果有红旗或该格处理过就不对该格进行任何判断*/
      return;
   mineNUM--;/*显示过数字或者空格的格子就表示多处理了一个格子,当所有格子都处理过了表示胜利*/
   if(Mine[i][j].roundnum==0&&Mine[i][j].num!=1)/*显示空格*/
   {
      DrawEmpty(i,j,1,7);
      Mine[i][j].num=0;
   }
   else
      if(Mine[i][j].roundnum!=0)/*输出雷数*/
      {
     DrawEmpty(i,j,0,8);
     sprintf(randmineNUM,"%d",Mine[i][j].roundnum);
     setcolor(RED);
     outtextxy(195+j*20,95+i*20,randmineNUM);
     Mine[i][j].num=0;/*已经输出雷数的格子用 0 表示已经用过这个格子*/
     return ;
      }
/*8 个方向递归显示所有的空白格子*/
   if(i!=0&&Mine[i-1][j].num!=1)
      ShowWhite(i-1,j);
   if(i!=0&&j!=9&&Mine[i-1][j+1].num!=1)
      ShowWhite(i-1,j+1);
   if(j!=9&&Mine[i][j+1].num!=1)
      ShowWhite(i,j+1);
```

```
    if(j!=9&&i!=9&&Mine[i+1][j+1].num!=1)
       ShowWhite(i+1,j+1);
    if(i!=9&&Mine[i+1][j].num!=1)
       ShowWhite(i+1,j);
    if(i!=9&&j!=0&&Mine[i+1][j-1].num!=1)
       ShowWhite(i+1,j-1);
    if(j!=0&&Mine[i][j-1].num!=1)
       ShowWhite(i,j-1);
    if(i!=0&&j!=0&&Mine[i-1][j-1].num!=1)
       ShowWhite(i-1,j-1);
}
void GamePlay(void)/*游戏过程*/
{
    int i,j,Num;/*Num 用来接收统计函数返回一个格子周围有多少地雷*/
    for(i=0;i<10;i++)
       for(j=0;j<10;j++)
      Mine[i][j].roundnum=MineStatistics(i,j);/*统计每个格子周围有多少地雷*/
    while(!kbhit())
    {
       if(LeftPress())/*鼠标左键盘按下*/
       {
      MouseGetXY();
      if(MouseX>280&&MouseX<300&&MouseY>65&&MouseY<85)/*重新来*/
      {
         MouseOff();
         gameAGAIN=1;
         break;
      }
      if(MouseX>190&&MouseX<390&&MouseY>90&&MouseY<290)/*当前鼠标位置在格子范围内
*/
      {
         j=(MouseX-190)/20;/*x 坐标*/
         i=(MouseY-90)/20;/*y 坐标*/
         if(Mine[i][j].flag==1)/*如果格子有红旗则左键无效*/
            continue;
         if(Mine[i][j].num!=0)/*如果格子没有处理过*/
         {
            if(Mine[i][j].num==1)/*鼠标按下的格子是地雷*/
            {
           MouseOff();
           GameOver();/*游戏失败*/
           break;
            }
            else/*鼠标按下的格子不是地雷*/
```

```
            {
          MouseOff();
          Num=MineStatistics(i,j);
          if(Num==0)/*周围没地雷就用递归算法来显示空白格子*/
             ShowWhite(i,j);
          else/*按下格子周围有地雷*/
          {
             sprintf(randmineNUM,"%d",Num);/*输出当前格子周围的雷数*/
             setcolor(RED);
             outtextxy(195+j*20,95+i*20,randmineNUM);
             mineNUM--;
          }
           MouseOn();
           Mine[i][j].num=0;/*点过的格子周围雷数的数字变为 0 表示这个格子已经用过
*/
           if(mineNUM<1)/*胜利了*/
           {
          GameWin();
          break;
           }
        }
     }
      }
   }
   if(RightPress())/*鼠标右键键盘按下*/
   {
      MouseGetXY();
      if(MouseX>190&&MouseX<390&&MouseY>90&&MouseY<290)/*当前鼠标位置在格子范围
内*/
      {
     j=(MouseX-190)/20;/*x 坐标*/
     i=(MouseY-90)/20;/*y 坐标*/
     MouseOff();
     if(Mine[i][j].flag==0&&Mine[i][j].num!=0)/*本来没红旗现在显示红旗*/
     {
        DrawRedflag(i,j);
        Mine[i][j].flag=1;
     }
     else
        if(Mine[i][j].flag==1)/*有红旗标志再按右键就红旗消失*/
        {
           DrawEmpty(i,j,0,8);
           Mine[i][j].flag=0;
        }
```

```
        }
        MouseOn();
        sleep(1);
        }
    }
}
```