

报告名称：扫雷综合报告

高程1班

信09

1953729

吴浩泽



完成日期：2021年1月2日

1. 题目

完成游戏扫雷的实现

菜单一

选择难度并显示内部数组

菜单二

输入初始位置并显示被打开的初始区域

菜单三

内部数组基础版

菜单四

内部数组完整版

在3的基础上，增加三个特殊输入，分别用于显示本局游戏已运行时间、标记某位置为雷（无论该位置真实情况是否为雷，等同于参考游戏的鼠标右键功能）、取消标记

菜单五

画出伪图形化的框架并显示内部数据

菜单六

在伪图形化的框架上移动鼠标，判断鼠标的位置

菜单七

用鼠标在伪图形化的框架上单击初始位置并显示被打开的初始区域

菜单八

伪图形化游戏基础版

菜单九

伪图形化游戏完整版在8 的基础上，可显示剩余雷数、计算本局游戏已运行时间（空格显示时间、游戏结束时显示时间）

从头开始想这道题，肯定是有难度的，那么可以一步一步去分析。整道题是积少成多，不是一次就能都想出来的，所以将其分块才是最正确的处理方式。

首先先确定字体大小和cmd窗口大小以及缓冲区大小。

```
cct_setcolor(0, 7);
```

之后调用了menu菜单函数，每次选项结束后都会重新清屏并重新罗列菜单。

接下来再看56789都用到了图形化，所以将它们放到一个函数，伪图形函数。

那么最后的分类结果就是这样的。

```
while (1)
{
    if (num == 0)
    {
        cout << '0';
        break;
    }
    if (num == 1 || num == 2 || num == 3 || num == 4)
    {
        cct_cls();
        nanduxuanze = randu();
        cct_cls();
        neibushuzu(nanduxuanze, num);
    }
    if (num == 5 || num == 6 || num == 7 || num == 8 || num == 9)
    {
        cct_cls();
        nanduxuanze = randu();
        cct_cls();
        weituxing(nanduxuanze, num);
    }
    end(num, nanduxuanze);
}
```

当然了要把它们放到一个循环里，直到按0才会退出循环从而结束程序。

而这里的nandu函数是为了选择难度，把值给到neibushuzu函数或者weituxing函数

因为nandu函数从1到9都要用到，所以每一次都要调用它，把它放到工具cpp中，也是要有返回值，返回值给到nanduxuanze变量。

因为不同的菜单有可能cmd窗口或者字体出现差异，所以我把这些放在了end函数，end作为每一次的结束，起到的主要功能是按回车键结束，但是也加入了其他做法。

```
int end(int& num, int nanduxuanze)
{
if (num == 1 || num == 2 || num == 3 || num == 4)
    cout << "按回车键继续。";
```

```

if (num == 5 || num == 6 || num == 7 || num == 8 || num == 9)
{
    if (nanduxuanze == 1)
    {
        cct_gotoxy(0, 33);
        cout << "按回车键继续...";
    }
    if (nanduxuanze == 2 || nanduxuanze == 3)
    {
        cct_gotoxy(0, 54);
        cout << "按回车键继续...";
    }
}
while (_getch() != '\r')
;
cct_cls();
cct_setfontsize("新宋体", 24);
cct_setconsoleborder(100, 30, 100, 30);
num = menu();
return num;
}

```

这里每次都恢复到默认的100, 30, 100, 30, 都恢复到新宋体, 24号字, 每次结束以后再次调用menu函数, 确保每次都能循环之前。而需要注意的是end函数里面的num一定要传引用, 因为形参改变, 实参也要改变。而实际上到这里main函数就结束了, 具体的操作都放到了两个模块中

一个是neibushuzu函数

一个是weituxing函数

```

int menu();
int nandu();
int end(int&, int);
void yanse(char x);
void input(int&, int&, int, char dakai[][11], char biaoji[][11]);
void neibushuzu(int, int);
void nandul(int, int);
void kuosan(char saolei[][11], char fugai[][11], char dakai[][11], char biaoji[][11], int alph1, int alph2);
void output(int num, char saolei[][11], char fugai[][11], char biaoji[][11]);
void game1(char saolei[][11], char fugai[][11], char dakai[][11], char biaoji[][11], int alph1, int alph2, int num, int nanduxuanze, clock_t start, clock_t end);
void weituxing(int nanduxuanze, int num);
void kuangjial();
void difficulty1(int, int);
void shuchu(int num, char saolei[][11], char fugai[][11], char biaoji[][11]);
void shubiao(int num, int& flag, int& leishu, int& alph1, int& alph2, char dakai[][11], char biaoji[][11]);
void game2(char saolei[][11], char fugai[][11], char dakai[][11], char biaoji[][11], int alph1, int alph2, int leishu, int num, int nanduxuanze, clock_t start, clock_t end);

```

因为用到了函数重载, 主要是当时想的时候第一个想的三个嘛, 就直接函数重载了, 如果说要是更多种难度, 那么应该可以用到一个数组选部分的做法, 那么以上是我用到的函数, 省去了一些函数重载的相同类型函数。

那么整体设计思路大致就是这样, 把具体的操作都放到了neibushuzu函数的调用, 以及weituxing函数的调用。

3. 主要功能的实现

接下来大致介绍各个函数的设计思路

- ❖ menu函数顾名思义，就是菜单函数，是为了完成菜单选项，并且给出返回值，再在main函数里根据菜单函数的返回值确定以后的每一步应该调用什么函数。

那么我先说明一点，由于我是函数重载做的，所以以下的函数实现，我都是以一个为例去说明。

- ❖ 为了篇幅，我就不写三个函数了。

kuosan函数是整个项目里唯一的递归函数，它的作用就是递归来正确完成程序的执行，而递归的思想也就是让一个位置是0的情况下，去周围8个位置找雷，如果有0则继续扩散，而一定要注意的是扩散的条件，绝不能来回扩散，也就是说我扩散了一次，之后下一个扩散的时候又扩散回来，就会递归死循环，一直递归，这样函数出不来的，所以这个时候我就引进了dakai数组，dakai数组的目的是如果有被kuosan的也就是被dakai了，那么我下次就不扩散对应位置的元素，这个情况的话我是不用考虑边界的，因为我的数组比原来都大了一圈，也就是我当时赋初值的时候赋的是'\0'而不是字符0，这样我扩散到边界的时候自然，它就不再扩散了，这就是多一圈数组的好处，而且我的1到16还不用对应0到15，这样好看多了。

```
void kuosan(char saolei[][11], char fugai[][11], char dakai[][11], char biaoji[][11], int alph1, int alph2)
{
    for (int i = alph1 - 1; i <= alph1 + 1; i++)
    {
        for (int j = alph2 - 1; j <= alph2 + 1; j++)
        {
            if (dakai[i][j] == 1)
            {
                if (biaoji[i][j] == 1)
                    fugai[i][j] = 'X';
                else
                    fugai[i][j] = saolei[i][j];
                continue;
            }
            if (dakai[i][j] == 0)
            {
                if (biaoji[i][j] == 1)
                    fugai[i][j] = 'X';
                else
                    fugai[i][j] = saolei[i][j];
                dakai[i][j] = 1;
            }
            if (saolei[i][j] == '0')
                kuosan(saolei, fugai, dakai, biaoji, i, j);
        }
    }
    for (int i = 1; i <= 9; i++)
    {
        for (int j = 1; j <= 9; j++)
        {
            if (fugai[i][j] == '\0')
                fugai[i][j] = 'X';
        }
    }
}
```

其实本质也是属于把覆盖数组用扫雷数组赋值，其余元素X赋值，最后输出的是覆盖数组。接下来neibushuzu函数以及weituxing函数要分步选择该调用哪个函数，那么就会分为nandu1函数，nandu2函数，nandu3函数，伪图形的话就分为difficulty1，difficulty2，difficulty3函数。

对于nandu1函数来说，先设置时间

```
clock_t start, end;
start = clock();
end = clock();
```

然后设置数组

```
char saolei[11][11] = {};
char fugai[11][11] = {};
char dakai[11][11] = {};
char biaoji[11][11] = {};
```

输入的话设置一下alpha1和alpha2，进行传参，那么之后就是分情况分为num为2或者3，num为4的情况。

```
for (int i = 1; i <= 9; i++)
    for (int j = 1; j <= 9; j++)
        if (biaoji[i][j] == 2)
            biaoji[i][j] = 0;
```

然后把biaoji数组从2放置到0，之后生成雷。

之后12分别对应直接调用kuosan和output函数，4的话因为比较复杂，所以再重新设置一个函数game1

然后说明一下game1的实现，game1实现过后其实程序就完事了，

game1其实是重复第一次的操作，也就是0就扩散，不是0加一个直接打开，是雷就结束，如果fugai数组里的元素的X对应的都是扫雷数组里的雷，那么胜利，也就是找到了所有的雷。

那么相对应的输入函数以及输出函数也有对应的实现。

伪图形的话，会发现，首先是框架，框架都是一样的，然后改变的就是对应位置的颜色以及数字，那么给出框架函数，然后再给出color函数，再给出shuchu函数，把每一块都看成一个整体，因为每次改变都是一个小部分改变的。

同样相对应的也是重新来一个difficulty1，2，3函数，代替原来的nandu1，2，3函数，然后8和9比较复杂，那么就来了一个game2函数，与之前的想法是一样的，然后相对应输入函数input，这里重新来一个shubiao函数，用shubiao函数内部进行输入。

那么鼠标函数也是有讲究的，

```
for (int i = 1; i <= 9; i++) {
    for (int j = 1; j <= 9; j++) {
        if (X >= 6 * j - 2 && X <= 6 * j + 1 && Y >= 3 * i && Y <= 3 * i + 1) {
            alph1 = i; alph2 = j; t = 1;
            if (dakai[alph1][alph2] == 0 && biaoji[alph1][alph2] != 1) {
                biaoji[alph1][alph2] = 1; leishu--;
            }
        }
        else if (biaoji[alph1][alph2] == 1) {
            biaoji[alph1][alph2] = 2; leishu++;
        }
    }
}
```

这样一个范围，就可以计算出i和j从鼠标，连接到坐标，再连接到数组的对应的元素，其次，注意一下鼠标的循环应用，何时出循环，都是自行判断的慢功夫。那么最后第9个还会用到shengyuleishu的函数，注意传的也是引用，还是那句话，形参改变，我想让它真正改变，那么就指针，能用引用就不用指针，基本就是这样。

The image is a collage of handwritten notes and diagrams on lined paper, likely from a notebook or a collection of papers. The notes are written in Chinese and English, and the diagrams are hand-drawn on grid paper.

Top Left Page:

- A coordinate system with x and y axes. The x-axis is labeled 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. The y-axis is labeled 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. The origin is labeled (0,0).
- Points are plotted and labeled: (0,0) is '黑' (black), (1,0) is '蓝' (blue), (2,0) is '绿' (green), (3,0) is '青' (cyan), (4,0) is '红' (red), (5,0) is '粉' (pink), (6,0) is '黄' (yellow), (7,0) is '白' (white), (8,0) is '亮黑' (bright black), (9,0) is '亮蓝' (bright blue), (10,0) is '亮绿' (bright green), (11,0) is '亮青' (bright cyan), (12,0) is '亮红' (bright red), (13,0) is '亮粉' (bright pink), (14,0) is '亮黄' (bright yellow), (15,0) is '亮白' (bright white).
- A box labeled 'cct_setColor' is drawn.

Top Right Page:

- A coordinate system with x and y axes. The x-axis is labeled 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. The y-axis is labeled 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. The origin is labeled (0,0).
- Points are plotted and labeled: (0,0) is '黑' (black), (1,0) is '蓝' (blue), (2,0) is '绿' (green), (3,0) is '青' (cyan), (4,0) is '红' (red), (5,0) is '粉' (pink), (6,0) is '黄' (yellow), (7,0) is '白' (white), (8,0) is '亮黑' (bright black), (9,0) is '亮蓝' (bright blue), (10,0) is '亮绿' (bright green), (11,0) is '亮青' (bright cyan), (12,0) is '亮红' (bright red), (13,0) is '亮粉' (bright pink), (14,0) is '亮黄' (bright yellow), (15,0) is '亮白' (bright white).
- A box labeled 'cct_setColor' is drawn.

Bottom Left Page:

- Handwritten text: 'cct_cls()' followed by '清除' (clear).
- Handwritten text: 'cct_setcolor(背景, 前景)' (cct_setcolor(background, foreground)).
- Handwritten text: 'cct_gotoxy(x, y)'.
- Handwritten text: 'cct_getxy(x, y)'.
- Handwritten text: 'cct_setconsoleborder(列数, 行数, 后列数, 后行数)' (cct_setconsoleborder(column count, row count, back column count, back row count)).
- Handwritten text: 'cct_showch(x, y, 字符, 背景, 前景, 次数)' (cct_showch(x, y, character, background, foreground, count)).
- Handwritten text: 'cct_setfontsize("包阵", 高, 宽)' (cct_setfontsize("array", height, width)).

Bottom Right Page:

- A coordinate system with x and y axes. The x-axis is labeled 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. The y-axis is labeled 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. The origin is labeled (0,0).
- Points are plotted and labeled: (0,0) is '黑' (black), (1,0) is '蓝' (blue), (2,0) is '绿' (green), (3,0) is '青' (cyan), (4,0) is '红' (red), (5,0) is '粉' (pink), (6,0) is '黄' (yellow), (7,0) is '白' (white), (8,0) is '亮黑' (bright black), (9,0) is '亮蓝' (bright blue), (10,0) is '亮绿' (bright green), (11,0) is '亮青' (bright cyan), (12,0) is '亮红' (bright red), (13,0) is '亮粉' (bright pink), (14,0) is '亮黄' (bright yellow), (15,0) is '亮白' (bright white).
- A box labeled 'cct_setColor' is drawn.

前面4个主要是想法，从第五个开始主要是算坐标，56789是坐标和老师给的鼠标函数的综合应用。最后可以找到规律，坐标和数组对应的元素是有规律的，然后老师给的鼠标的函数实际上就等同于输入函数。取坐标的话，可以通过鼠标的范围，逆着从范围中找到数组对应的行和列，i和j。

4. 调试过程碰到的问题

调试过程的确很让人厌烦，有些 bug 改了好长时间也修复不了，然后晕头转向，都不知道自己干嘛，而且有时候调试过程太缓慢了，其实主要还是第一次的想法很关键，想法要周全，要不然根本找不到 bug，改起来巨难受，比如这次的 kuosan 函数，当时我就卡了一上午，最后发现问题是我选完第一块，去扩散剩余8个的时候，我从剩余8个里面的一个扩散之后又回到了我这里面，就造成了递归函数的死循环，而且关键是一直没考虑到这个点，调试之后也不知道哪里出错了，反正到kuosan函数就异常，但是表面看上去还是挺对的，所以有时候就还是思维没达到，想法不够周全。

❖

5. 心得体会

❖

❖

❖

❖

❖

❖

❖

❖

❖

本次作业很大，内容很多，收获也很多，而且非常锻炼逻辑思维，为以后的逻辑编程打下了基础，那么我来具体说说都增长了哪些知识。

在这种较为复杂的程序中，同一个功能一定要分函数来实现，而且函数的细微差别可以靠参

数的不同而改变。而且一定要统筹兼顾，做前面考虑后面，要尽量使后面的能用到前面的，在初始写的时候给参数留下空间，以便于后面加进来，改动不会太大，方便后续操作。

如果要是函数重载的话，长记性了，下次就应该先从头搞完第一个，然后搞完之后，再复制粘贴，然后改变对应参数，这样就会简单很多，要不然改一个就得改三个，很麻烦。

6. 附件：源程序

```

void neibushuzu(int nanduxuanze, int num)
{
    if (nanduxuanze == 1)
        nandu1(nanduxuanze, num);
    else if (nanduxuanze == 2)
        nandu2(nanduxuanze, num);
    else
        nandu3(nanduxuanze, num);
}

void kuosan(char saolei[][11], char
fugai[][11], char dakai[][11], char
biaoji[][11], int alph1, int alph2)
{
    for (int i = alph1 - 1; i <= alph1 + 1; i++)
    {
        for (int j = alph2 - 1; j <= alph2 + 1; j++)
        {
            if (dakai[i][j] == 1)
            {
                if (biaoji[i][j] == 1)
                    fugai[i][j] = 'X';
                else
                    fugai[i][j] = saolei[i][j];
                continue;
            }
            if (dakai[i][j] == 0) {
                if (biaoji[i][j] == 1)
                    fugai[i][j] = 'X';
                else
                    fugai[i][j] = saolei[i][j];
                dakai[i][j] = 1;
            }
            if (saolei[i][j] == '0')
                kuosan(saolei, fugai, dakai, biaoji, i, j);
        }
    }
    for (int i = 1; i <= 9; i++)
    {
        for (int j = 1; j <= 9; j++)
        {
            if (fugai[i][j] == '\0')
                fugai[i][j] = 'X'
        }
    }
}

void nandu1(int nanduxuanze, int num)
{
    clock_t start, end;
    start = clock();
    end = clock();
    cout << "内部数组: " << endl;
    cout << " | 1 2 3 4 5 6 7 8 9" << endl;
    cout << "-----" <<
endl;
    char saolei[11][11] = {};
    char fugai[11][11] = {};
    char dakai[11][11] = {};
    char biaoji[11][11] = {};
    int alph1, alph2;
    if (num == 2 || num == 3)
    {
        for (int i = 1; i <= 9; i++)
        {
            char alph = 64;
            cout << (char)(alph + i)
<< " |";
            cout << "X X X X X X X X
X" << endl;
        }
        cout << endl << endl;
        cout << "输入非雷位置的行列坐标
(先行后列, 严格区分大小写, 例: G1/Af, 按Q/q退出): ";
        input(alph1, alph2, num, dakai,
biaoji);
        if (alph1 == 'q' || alph1 ==
'Q')
        {
            cout << endl;
            return;
        }
        cout << endl << endl;
    }
    if (num == 4)
    {
        for (int i = 1; i <= 9; i++)
        {
            char alph = 64;
            cout << (char)(alph + i)
<< " |";
            cout << "X X X X X X X X
X" << endl;
        }
        cout << endl << endl;
        cout << "特殊输入说明: & - 游戏
已运行时间(单字符即可, 不需要加坐标)" << endl;
    }
}

```

装

订

线

```
cout << "          ! - 标记该坐标为雷
(例: !E3)" << endl;
        cout << "          # - 取消
标记      (例: #E3)" << endl;
        cout << "请输入（坐标必须先行后
列，严格区分大小写，例: G1/Af，按Q/q退出）: ";
        input(alph1, alph2, num, dakai,
biaoji);
        while (alph1 == '&')
        {
            cout << endl;
            end = clock();
            cout << "已运行时间: "
<< (double)(end - start) / CLOCKS_PER_SEC << "
秒" << endl << endl;
            cout << "特殊输入说明: &
- 游戏已运行时间(单字符即可，不需要加坐标)" <<
endl;
            cout << "          !
- 标记该坐标为雷(例: !E3)" << endl;
            cout << "          #
- 取消标记      (例: #E3)" << endl;
            cout << "请输入（坐标必
须先行后列，严格区分大小写，例: G1/Af，按Q/q退
出）: ";
            input(alph1, alph2, num,
dakai, biaoji);
        }
        if (alph1 == 'q' || alph1 ==
'Q')
        {
            cout << endl;
            return;
        }
        cout << endl << endl;
        while (biaoji[alph1][alph2] !=
0)
        {
            cout << "当前数组: " <<
endl;
            cout << " | 1 2 3 4 5 6
7 8 9" << endl;
            cout << "-----
-----" << endl;
            for (int i = 1; i <= 9;
i++)
            {
                char alph = 64;
                cout <<
(char)(alph + i) << " |";
                for (int j = 1;
j <= 9; j++)
                {
                    if
(biaoji[i][j] == 1)
```

```
cct_setcolor(4, 7);

        cout << 'X';

        cct_setcolor(0, 7);

        cout << ' ';

            }
        else

        cout << 'X' << ' ';

            }
        cout << endl;

        cout << endl << endl;
        cout << "特殊输入说明: &
- 游戏已运行时间(单字符即可，不需要加坐标)" <<
endl;
        cout << "          !
- 标记该坐标为雷(例: !E3)" << endl;
        cout << "          #
- 取消标记      (例: #E3)" << endl;
        cout << "请输入（坐标必
须先行后列，严格区分大小写，例: G1/Af，按Q/q退
出）: ";
        input(alph1, alph2, num,
dakai, biaoji);
        while (alph1 == '&')
        {
            cout << endl;
            end = clock();
            cout << "已运行
时间: " << (double)(end - start) /
CLOCKS_PER_SEC << "秒" << endl << endl;
            cout << "特殊输
入说明: & - 游戏已运行时间(单字符即可，不需要加
坐标)" << endl;
            cout <<
          ! - 标记该坐标为雷(例: !E3)" <<
endl;
            cout << "
          # - 取消标记      (例: #E3)" << endl;
            cout << "请输入
（坐标必须先行后列，严格区分大小写，例: G1/Af，
按Q/q退出）: ";
            input(alph1,
alph2, num, dakai, biaoji);
        }
        if (alph1 == 'q' ||
alph1 == 'Q')
        {
            cout << endl;
            return;
        }
        cout << endl << endl;
```

装

订

线

```
void yanse(char x)
{
    switch (x)
    {
        case '0':
            cct_setcolor(14, 0);
            break;
        case '1':
            cct_setcolor(14, 1);
            break;
        case '2':
            cct_setcolor(14, 2);
            break;
        case '3':
            cct_setcolor(14, 3);
            break;
        case '4':
            cct_setcolor(14, 4);
            break;
        case '5':
            cct_setcolor(14, 5);
            break;
        case '6':
            cct_setcolor(14, 6);
            break;
        case '7':
            cct_setcolor(14, 7);
            break;
        case '8':
            cct_setcolor(14, 8);
            break;
        case 'X':
            cct_setcolor(0, 7);
            break;
        case '*':
            cct_setcolor(0, 7);
            break;
    }
    cout << x;
    cct_setcolor(0, 7);
    cout << ' ';
}
```

```
void input(int& alph1, int& alph2, int num,
char dakai[][11], char biaoji[][11])
{
    while (1)
    {
        alph1 = _getch();
        if (num == 4)
        {
            if (alph1 == '&')
            {
                cout << '&';
                return;
            }
            if (alph1 == '!')
            {
                cout << '!';
                while (1)
                {
                    alph1 =
                        _getch();
                    if
                        (alph1 >= 'A' && alph1 <= 'I')
                    {
                        cout << (char)(alph1);
                        alph1 = alph1 - 64;
                        break;
                    }
                }
            }
            while (1)
            {
                alph2 =
                    _getch();
                if
                    (alph2 >= '1' && alph2 <= '9')
                {
                    cout << (char)(alph2);
                    alph2 = alph2 - 48;
                    break;
                }
            }
            if
                (dakai[alph1][alph2] == 0)
            {
                biaoji[alph1][alph2] = 1;
                return;
            }
            if (alph1 == '#')
            {
                cout << '#';
            }
        }
    }
}
```

装订线

```

void output(int num, char saolei[][11], char fugai[][11], char biaoji[][11])
{
    if (num == 1)
    {
        for (int i = 1; i <= 9; i++)
        {
            int alph = 64;
            cout << (char)(alph + i) << " |";
            for (int j = 1; j <= 9; j++)
            {
                cout << saolei[i][j] << ' ';
            }
            cout << endl;
        }
        cout << endl << endl;
    }
    if (num == 2 || num == 3 || num == 4)
    {
        if (num == 2)
            cout << "点开后的数组: ";
        if (num == 3 || num == 4)
            cout << "当前数组: ";
        cout << " | 1 2 3 4 5 6 7 8 9" << endl;
        for (int i = 1; i <= 9; i++)
        {
            int alph = 64;
            cout << (char)(alph + i) << " |";
            for (int j = 1; j <= 9; j++)
            {
                if (biaoji[i][j] == 1)
                {
                    cct_setcolor(4, 7);
                    cout << fugai[i][j];
                    cct_setcolor(0, 7);
                    cout << ' ';
                }
            }
        }
    }
}

void game1(char saolei[][11], char fugai[][11], char dakai[][11], char biaoji[][11], int alph1, int alph2, int num, int nanduxuanze, clock_t start, clock_t end)
{
    kuosan(saolei, fugai, dakai, biaoji, alph1, alph2);
    output(num, saolei, fugai, biaoji);
    while (saolei[alph1][alph2] != '*' || biaoji[alph1][alph2] != 0)
    {
        if (num == 3)
            cout << "输入非雷位置的行列坐标（先行后列，严格区分大小写，例：G1/Af，按Q/q退出）: ";
        if (num == 4)
        {
            cout << "特殊输入说明: &
- 游戏已运行时间(单字符即可，不需要加坐标)" << endl;
            cout << "          !
- 标记该坐标为雷(例: !E3)" << endl;
            cout << "          #
- 取消标记      (例: #E3)" << endl;
            cout << "请输入（坐标必须先先行后列，严格区分大小写，例：G1/Af，按Q/q退出）: ";
        }
        input(alph1, alph2, num, dakai, biaoji);
        while (alph1 == '&')
        {
            cout << endl;
            end = clock();
            cout << "已运行时间: " << (double)(end - start) / CLOCKS_PER_SEC << "秒" << endl << endl;
            cout << "特殊输入说明: &
- 游戏已运行时间(单字符即可，不需要加坐标)" << endl;
            cout << "          !
- 标记该坐标为雷(例: !E3)" << endl;
            cout << "          #
- 取消标记      (例: #E3)" << endl;
            cout << "请输入（坐标必须先先行后列，严格区分大小写，例：G1/Af，按Q/q退出）: ";
            input(alph1, alph2, num, dakai, biaoji);
        }
    }
}
    
```

装
订
线

```
void weituxing(int nanduxuanze, int num)
{
    if (nanduxuanze == 1)
        difficulty1(nanduxuanze, num);
    else if (nanduxuanze == 2)
        difficulty2(nanduxuanze, num);
    else
        difficulty3(nanduxuanze, num);
}

void shengyuleishu(int leishu, int num)
{
    if (num == 9)
    {
        cct_gotoxy(32, 0);
        if (leishu > 0)
            cout << setw(2) <<
leishu << " ";
        else
            cout << setw(2) << 0 <<
" ";
    }
}

void shuchu(int num, char saolei[][11], char
fugai[][11], char biaoji[][11])
{
    if (num == 5 || num == 6)
    {
        for (int i = 1; i <= 9; i++)
        {
            for (int j = 1; j <= 9;
j++)
            {
                cct_gotoxy(6 * j
- 2, 3 * i);
                cct_setcolor(7,
7);
                cout << " ";
                cct_gotoxy(6 * j
- 2, 3 * i + 1);
                cout << " ";

                color(saolei[i][j]);
                cct_setcolor(7,
7);
                cout << " ";

            }
        }
    }
    if (num == 7 || num == 8 || num == 9)
```

```
void color(char x)
{
    switch (x)
    {
        case '0':
            cct_setcolor(7, 7);
            break;
        case '1':
            cct_setcolor(7, 1);
            break;
        case '2':
            cct_setcolor(7, 2);
            break;
        case '3':
            cct_setcolor(7, 3);
            break;
        case '4':
            cct_setcolor(7, 4);
            break;
        case '5':
            cct_setcolor(7, 5);
            break;
        case '6':
            cct_setcolor(7, 6);
            break;
        case '7':
            cct_setcolor(7, 7);
            break;
        case '8':
            cct_setcolor(7, 8);
            break;
        case 'X':
            cct_setcolor(6, 6);
            break;
        case '*':
            cct_setcolor(7, 0);
            break;
    }
    if (x == '0' || x == 'X')
        cout << ' ';
    else
        cout << x;
    cct_setcolor(0, 7);
}
```

```

void shubiao(int num, int& flag, int& leishu,
int& alph1, int& alph2, char dakai[][11], char
biaoji[][11])
{
    int X = 0, Y = 0;
    int ret, maction;
    int keycode1, keycode2;
    int loop = 1;
    cct_enable_mouse();
    cct_setcursor(CURSOR_INVISIBLE);
    while (loop)
    {
        ret =
cct_read_keyboard_and_mouse(X, Y, maction,
keycode1, keycode2);
        if (ret == CCT_MOUSE_EVENT)
        {
            cct_gotoxy(0, 30);
            cout << "[当前光标]";
            int t = 0;
            switch (maction) {
            case MOUSE_ONLY_MOVED:
                cct_gotoxy(11,
30);
                for (int i = 1;
i <= 9; i++)
                {
                    for (int
j = 1; j <= 9; j++)
                    {
                        if (X >= 6 * j - 2 && X <= 6 * j + 1 &&
Y >= 3 * i && Y <= 3 * i + 1)
                        {
                            t = 1;

                            char zimu = 'A';

                            cct_gotoxy(11, 30);

                            cout << (char)(zimu + i - 1) << "行" <<
j - 1 << "列 ";

                            break;
                        }
                    }
                }
                if (t == 0)
                {
                    cct_gotoxy(11, 30);
                    cout << "
位置非法";
                }
            }
        }
    }
}

```