
# 实验报告 彩球游戏

信06 2352018 刘彦

2024/6/17

# 1. 题目及基本要求描述

## 1.1.内部数组，生成初始状态，寻找是否有初始可消除项

输入行数和列数后调用print_mark函数即可完成。

## 1.2.内部数组，消除初始可消除项后非0项下落并用0填充

调用base_remove函数，达到消除初始可消除项后，将消除位置的值变为0并将其上移，并用新值填充的目的。

## 1.3.内部数组，消除初始可消除项后查找消除提示

调用base_remove函数，达到消除初始可消除项后，将消除位置的值变为0并将其上移，并用新值填充后，可以标出查找消除提示项的目的。

## 1.4.n*n的框架(无分隔线)，显示初始状态

## 1.5.n*n的框架(有分隔线)，显示初始状态

在输入row和col后，打印伪图形界面下的无/有分隔线边框和初始状态。

## 1.6.n*n的框架(无分隔线)，显示初始状态及初始可消除项

在输入row和col后，打印伪图形界面下的无分隔线边框和初始状态并显示初始可消除项。

## 1.7.n*n的框架(有分隔线)，消除初始可消除项后显示消除提示

在输入row和col后，打印无分隔线边框和初始状态并显示初始可消除项和消除提示。

## 1.8.cmd图形界面完整版(有分隔线，鼠标移动时显示坐标，右键退出)

在之前的基础上，加上鼠标的功能，在移动到彩球上时显示坐标，左键可以选择有消除提示的彩球，右键退出游戏。

## 1.9.cmd 图形界面完整版

在之前的基础上，加上所有的功能，鼠标左键选择两个相邻的提示项交换后进行消除，并获得相应得分，若交换后不可消去，则给出提示后不交换，右键退出游戏。

# 2. 整体设计思路

## 2.1.重要函数

### 2.1.1. 汇总函数ball_base_all

汇总整个函数的输出。

### 2.1.2. draw_nine函数

汇总第9步的实现。

### 2.1.3. 设置函数base_remove

通过检测矩阵中的零值，将它们标记并消除，然后让上方的非零元素下移以填补空缺。在用户按下回车键后，函数会在消除后的零位置随机填充新的数值，并通过控制台输出实时更新矩阵状态，使用不同颜色高亮显示零值位置，直到没有更多零值可以消除为止。

### 2.1.4. 检验函数

[1] base_check0函数

检查在数组中指定位置的元素是否能够被消除，即判断该元素在水平或垂直方向上是否有至少两个相同的相邻元素，从而满足消除条件。如果存在至少三个连续相同的元素，函数返回1，表示可以消除；否则，返回0，表示不可消除。

[2] base_check1函数

遍历整个二维数组，使用base_check0函数检查数组中的每个元素是否满足消除条件。如果发现数组中存在任何可以消除的项（即至少有三个连续相同的元素），则函数计数器num会增加。在遍历结束后，如果num不为零，表示找到了初始可消除项；如果num为零，表示没有找到可消除项。

[3] base_check2函数

检查在二维数组中，指定位置的元素如果与相邻元素（上下左右）交换后，是否能够形成至少一组满足消除条件的元素（即至少三个相同元素连续相邻）。函数通过模拟交换操作并检查交换后是否能够触发消除，如果存在至少一种交换能够导致消除，函数返回1；如果所有可能的交换都不能形成消除，函数返回0。

## 2.2.总体思路

### 2.2.1. 初始化与准备

函数开始时，初始化必要的变量，包括棋盘的备份数组arrayCopy0，用于在游戏过程中保存和恢复原始棋随机数生成和棋盘初始化。使用srand和rand函数填充棋盘数组array，每个元素为1到9之间的随机整数。print_origin函数打印棋盘的初始状态，包括行和列的标记。

### 2.2.2. 界面绘制

使用控制台操作函数如cct_cls()和cct_setconsoleborder()来准备游戏界面，包括清屏和设置控制台边框。绘制游戏的初始棋盘状态，包括棋盘边框和初始元素。

### 2.2.3. 游戏逻辑处理

通过base_check1函数检查棋盘上的元素是否可以进行下落操作。如果可以，调用draw_fall函数模拟元素下落。如果元素不能下落，直接在棋盘上绘制静态元素，并标记可交换的元素。

### 2.2.4. 游戏环节

启用鼠标输入，并设置鼠标光标为不可见，允许玩家选择棋盘上的元素，并根据选择执行交换操作。

当玩家选择两个元素并执行交换时，函数会检查交换是否有效。如果有效，更新棋盘状态，并根据base_check2函数的结果判断是否满足消除条件。如果满足消除条件，执行消除操作，更新玩家分数，并重新绘制棋盘。

### 2.2.5. 游戏循环与结束条件

游戏循环持续进行，直到棋盘上没有可交换的元素组，此时游戏结束。在游戏结束时，显示最终分数，并提供重新开始或退出游戏的选项。

### 2.2.6. 重要数组的功能

[1] arrayCopy0

作用：arrayCopy0的主要作用是作为棋盘原始状态的备份。在游戏开始时，它被用来复制array数组中的数据，确保在任何交换或操作之前，都有一个可以恢复到原始状态的副本。

使用场景：当玩家进行操作（如选择和交换元素）之前，arrayCopy0被用来保存当前棋盘的状态。如果操作后发现没有形成可以消除的组合，或者操作不符合游戏规则，可以使用arrayCopy0来恢复到操作前的状态。

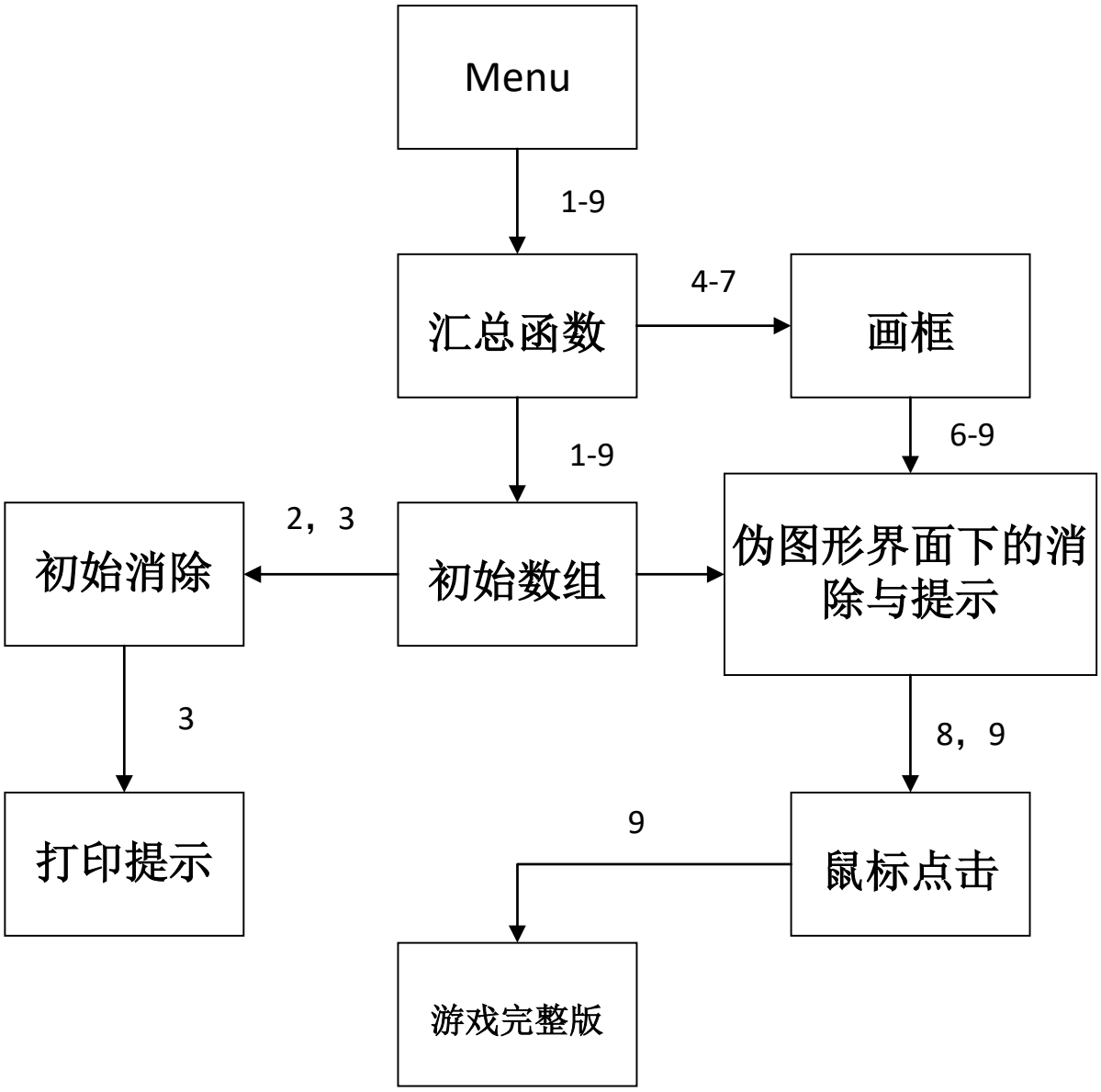
[2] arrayCopy

作用：arrayCopy则是用来在游戏过程中临时存储棋盘状态的数组。它允许在不直接修改原始棋盘状态array的情况下，进行各种尝试和计算。

使用场景：

在玩家选择元素并尝试交换时，arrayCopy被用来模拟交换后的状态，以检查交换是否有效或是否形成了可消除的组合。在执行消除操作后，arrayCopy可能被用来存储更新后的棋盘状态，然后再将这些更新反映到array中，以实现棋盘状态的动态更新。

## 3. 主要功能的实现




### 3.1. 检查可消除项

base_check0函数检查单个元素是否符合消除条件，即在水平或垂直方向上是否有至少两个相同的相

邻元素。base_check1函数遍历整个棋盘，使用base_check0来确定是否有任何可消除的项。

## 3.2.标记和提示可消除项

print_mark函数在控制台上高亮显示所有可消除的项。base_check2函数检查棋盘上任意两个相邻元素交换后是否能够形成可消除的组合，并在print_tips中显示这些提示。base_remove函数首先找到所有可消除的项，并将它们的位置记录下来。然后，它将这些项设置为0，并让上面的元素下落以填补空位。最后，它在棋盘顶部随机填充新的元素。

## 3.3.彩球的绘制与清除

根据提供的彩球位置 (i, j) 和颜色 color，以及边框类型 border，在控制台的相应位置绘制彩球。彩球可以是空心或实心，由 border 参数决定。

绘制游戏棋盘的边框时，根据 border 参数的不同，可以绘制简单的边框或带有额外分隔线的复杂边框。row 和 column 参数定义了棋盘的大小。

彩球下落与消除的时候，首先，它尝试找到并标记所有可消除的彩球，并将它们设置为0。然后，让上面的彩球下落以填补空缺。如果 have_score 为真，则计算并返回当前消除操作的得分。

在棋盘上特定位置清除彩球时，通常用于表示消除后的空位。通过在指定位置打印空格来实现。

## 3.4.用户交互和游戏循环

根据用户的选择，程序可能进入一个循环，等待用户输入或鼠标操作来执行特定的功能。使用_getch函数暂停程序，直到用户按下回车键。在8、9模式下，程序会启用鼠标输入，并根据用户的鼠标点击来选择和确认可消除的项。程序使用cct_enable_mouse启用鼠标输入，并设置鼠标光标为不可见。通过cct_read_keyboard_and_mouse函数读取鼠标位置和动作，响应用户的鼠标点击事件。

如果用户选择执行消除操作，程序会调用base_remove函数来识别和消除棋盘上的可消除项，然后更新棋盘状态。在循环中，根据棋盘状态的变化，程序可能需要重新打印棋盘，以反映最新的状态。游戏中使用 cct_gotoxy(x9, y9) 定位到控制台的特定位置，然后输出当前鼠标的位置信息，如 "[当前光标]"，以提供玩家光标位置的实时反馈。根据 maction 变量的值，代码会响应不同类型的鼠标事件。

当玩家在棋盘区域内使用鼠标左键点击时，游戏会检查点击的位置是否合法，并判断是否选择了可交换的彩球。如果选择合法，游戏会高亮显示选中的彩球，并允许玩家进行交换。用户可以通过特定的操作——鼠标右键来退出游戏循环。一旦退出条件被触发，loop变量会被设置为假，循环结束。

## 3.5.游戏循环与结束条件

游戏循环持续进行，直到棋盘上没有可交换的元素组，此时游戏结束。在游戏结束时，显示最终分

数，并提供重新开始或退出游戏的选项。

# 4. 调试过程碰到的问题

## 4.1.找寻输入提示的问题

输入提示的寻找也是此次程序的难点之一，一开始，我使用作业要求中的提示的思路尝试完成，很长时间都不能涵盖所有的情况，总是有缺漏，还堆叠了许多无用代码，为了解决这个问题，我尝试采用自己的思路，先将原数组复制一遍，再复制的数组中尝试交换，若交换后可消除，则标记为提示项的方法，有效地解决了问题。

## 4.2.彩球下落的问题

第一遍做彩球下落的过程时，我采用重复打印数组的方式，不但耗费了大量时间调试，而且下落效果不好，无法做到一个一个下落，后来换用另外一个逻辑，在下落过程中，用白色消除掉原有的彩球，然后再相应位置画出新的彩球，这样可以做到按列一个彩球一个彩球依次下落的效果。

## 4.3.游戏过程中出现循环的问题

在开始调试第9个菜单的时候，我发现在鼠标移动后会陷入长期循环，始终展示第一次鼠标移动时的消除结果，在长时间检查后，修改了多处逻辑后均没有奏效，当时本以为是draw_fall函数的判断是否跳出do-while循环的逻辑有误，后来偶然间发现，在实现时多建了一个没有用的for循环。这说明了写程序时一定要严谨且细心，否则容易反低级错误，还会耽误大量时间。

# 5. 心得体会

## 5.1.在做作业时要不断优化代码，使逻辑更加清晰

做作业时，要及时优化代码，努力使自己代码的逻辑和可读性更强，这样可以使自己更加熟悉代码，减少下一次阅读所需要的时间，从而提高效率，并减少出错，降低代码维护的时间。同时，一个清晰的架构设计有助于管理复杂的逻辑和模块关系。良好的架构可以提升代码的可维护性和扩展性，简化后期维护工作。

## 5.2.将一整个游戏拆分成多个步骤有利于逐层深入

在编写一个较复杂且功能较多的程序时，分步进行有利于初学者思考逐渐深入，从而能够较为容易的完成这样的程序。这样可以将每个小题循序渐进，逐步完成，减轻压力，而且还可以将程序分为多个

模块，增强了可读性，有助于分阶段开发和单独测试。

## 5.3. 充分利用命名、宏定义和注释

使用描述性的命名可以增加代码的可读性，并可以遵循一致的命名规范，如驼峰命名法或下划线命名法，以区分变量类型和作用域，还要避免模糊命名：避免使用模糊或通用的名称。

使用宏定义来表示常量值，有助于在多处使用时保持一致性并易于修改。

在规模较大的程序中，代码的逻辑可能比较复杂，所以对每个函数和模块一定要有清晰的注释。包含对函数和模块的用途和功能的描述，输入和输出的定义以及函数或模块之间的依赖关系等等。

## 5.4. 及时完成作业

拖延可能导致最后时刻匆忙完成作业，增加压力和焦虑，高程大作业的完成需要深入的理解和思考，无法在最后紧张的情况下较好的完成。所以，在完成较大型的程序时，要做好安排和规划，这样可以循序渐进，达到较好的效果。

# 6. 附件：源程序

```
/**************************************************************************
  函数名称: draw_nine
  功    能: 第9步的汇总函数
  输入参数:
  返 回 值:
  说    明:
**************************************************************************/
void draw_nine(int array[MIN_ROWS][MAX_ROWS], int rows, int cols, int choice)
{
                                        //这里定义变量采用双栏
    int a, b, c, d;                                 int lie;
     int x, y;                                      int directions[4][2] = { {-1, 0}, {1, 0}, {0, -1},
     int x9, y9;                                {0, 1} };   //方向数组
     int X = 0, Y = 0;   //鼠标位置                 int score = 0, score0 = 0;
     int ret, maction;                              int arrayCopy0[MAX_ROWS][MAX_COLS];
     int keycode1, keycode2;                        x9 = 0;
     int loop = 1;                                  y9 = 2 * rows + 2;
     char hang;
     for (int i = 0; i < rows; ++i)
     {
          for (int j = 0; j < cols; ++j)
               arrayCopy0[i][j] = array[i][j];
     }
     cct_cls();
     cct_getconsoleborder(a, b, c, d);
     cct_setcolor();
     cct_cls();
     d = 2 * cols + 5;
     c = rows * 4 + 4;
     cct_setconsoleborder(c + 25, d + 4);
     cct_gotoxy(0, 0);
     cct_setfontsize("新宋体", 28);
     cout << "屏幕: " << d << "行" << 40 << "列";
     cout << "(当前分数: " << score << ")";
     int arrayCopy[MAX_ROWS][MAX_COLS];

     draw_border(rows, cols, 1, choice);
     int count = 0;
     if (count == 0)
```

```
{
    for (int i = 0; i < rows; ++i)
    {
        for (int j = 0; j < cols; ++j)
            arrayCopy[i][j] = arrayCopy0[i][j];
    }
}


if (base_check1(arrayCopy0, rows, cols))
{
    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < cols; j++)
        {
            x = 4 * j + 2;
            y = 2 * i + 2;
            cct_gotoxy(x, y);
            cct_setcolor(arrayCopy0[i][j], COLOR_BLACK);
            if (base_check0(arrayCopy0, i, j, rows, cols))
                cout << "●";
            else
                cout << "○";
            cct_gotoxy(x, y + 1);
        }

    }
    while (1)
    {
        draw_fall(arrayCopy0, rows, cols, choice, arrayCopy, 0);
        for (int i = 0; i < rows; ++i)
        {
            for (int j = 0; j < cols; ++j)
                arrayCopy0[i][j] = arrayCopy[i][j];
        }
        if (base_check1(arrayCopy0, rows, cols))
            continue;
        else
            break;

    }

}
else
{
    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < cols; j++)
        {
            draw_ball(i, j, arrayCopy0[i][j], 1);
        }
    }
    int x0, y0;
    cout << endl;
    cct_getxy(x0, y0);
    cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
    for (int i = 0; i < rows; ++i)
    {
        for (int j = 0; j < cols; ++j)
        {
            // 检查并标记可消除项
            if (base_check2(arrayCopy0, directions, i, j, rows, cols))
            {
                draw_ball(i, j, arrayCopy0[i][j], 2);
            }

        }
    }
    cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
    cct_gotoxy(x0, y0);
}


int x_9, y_9;
int num0 = 1;
int color;
int i1, j1;
int i2, j2;
int count0 = 0;
cct_enable_mouse();
cct_setcursor(CURSOR_INVISIBLE);
while (loop)
```

```
    {
        ret = cct_read_keyboard_and_mouse(X, Y, maction, keycode1, keycode2);
        cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
        cct_gotoxy(x9, y9);
        cout << "[当前光标] ";

        if (ret == CCT_MOUSE_EVENT)
        {
            if (X <= 4 * cols && Y <= 2 * rows + 1)
            {
                hang = 'A' + Y / 2 - 1;
                lie = int(X / 4) + 1;
                if ((X % 4 == 2 || X % 4 == 3) && Y % 2 == 0 && Y > 0)
                {
                    cout << hang << "行" << lie << "列";

                    switch (maction)
                    {
                        case MOUSE_LEFT_BUTTON_CLICK:
                            if (base_check2(arrayCopy, directions, Y / 2 - 1, lie - 1, rows, cols))
                            {
                                if (num0 > 1)
                                {
                                    cct_gotoxy(x_9, y_9);
                                    cct_setcolor(color, COLOR_BLACK);
                                    cout << "◎";
                                }
                                cct_gotoxy(int(X / 2) * 2, Y);
                                x_9 = int(X / 2) * 2;
                                y_9 = Y;
                                color = arrayCopy[Y / 2 - 1][lie - 1];
                                cct_setcolor(color, COLOR_HWHITE);
                                cout << "◎";
                                Sleep(TIME * 20);
                                cct_gotoxy(x9, y9);
                                cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
                                cout << "当前选择" << hang << "行" << lie << "列";
                                Sleep(TIME * 20);
                                for (int i = 0; i < rows; ++i)
                                {
                                    for (int j = 0; j < cols; ++j)
                                        arrayCopy0[i][j] = arrayCopy[i][j];
                                }
                                if (num0 % 2 == 1)
                                {
                                    i1 = Y / 2 - 1;
                                    j1 = lie - 1;
                                }
                                if (num0 % 2 == 0)
                                {
                                    i2 = Y / 2 - 1;
                                    j2 = lie - 1;

                                    int tmp0;
                                    tmp0 = arrayCopy0[i1][j1];
                                    arrayCopy0[i1][j1] = arrayCopy0[i2][j2];
                                    arrayCopy0[i2][j2] = tmp0;

                                    if (base_check0(arrayCopy0, i1, j1, rows, cols) || base_check0(arrayCopy0,
i2, j2, rows, cols))
                                    {
                                        cct_gotoxy(x9, y9);
                                        cout << "交换" << char(i1 + 'A') << "行" << j1 + 1 << "列" << "<=>" <<
char(i2 + 'A') << "行" << j2 + 1 << "列";

                                        cct_gotoxy(x9, y9 + 1);

                                        int count = 0;
                                        if (count == 0)
                                        {
                                            for (int i = 0; i < rows; ++i)
                                            {
                                                for (int j = 0; j < cols; ++j)
                                                    arrayCopy[i][j] = arrayCopy0[i][j];
                                            }
                                        }


                                        if (base_check1(arrayCopy0, rows, cols))
                                        {
                                            for (int i = 0; i < rows; i++)
                                            {
                                                for (int j = 0; j < cols; j++)
```

```
                    {
                        x = 4 * j + 2;
                        y = 2 * i + 2;
                        cct_gotoxy(x, y);
                        cct_setcolor(arrayCopy0[i][j], COLOR_BLACK);
                        if (base_check0(arrayCopy0, i, j, rows, cols))
                            cout << "●";
                        else
                            cout << "○";
                        cct_gotoxy(x, y + 1);
                    }

                }
                while (1)
                {
                    score0 = draw_fall(arrayCopy0, rows, cols, choice, arrayCopy,
1);

                    score += score0;
                    for (int i = 0; i < rows; ++i)
                    {
                        for (int j = 0; j < cols; ++j)
                            arrayCopy0[i][j] = arrayCopy[i][j];
                    }
                    draw_fall(arrayCopy0, rows, cols, choice, arrayCopy, 1);
                    if (!base_check1(arrayCopy0, rows, cols))
                        break;

                }

                cct_gotoxy(0, 0);
                cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
                cout << "屏幕: " << d << "行" << 40 << "列";
                cout << "(当前分数: " << score << ")";
                for (int i = 0; i < rows; i++)
                {
                    for (int j = 0; j < cols; j++)
                    {
                        if (base_check2(arrayCopy0, directions, i, j, rows, cols))
                        {
                            count0++; // 如果找到满足条件的位置，增加计数
                        }
                    }
                }

                if (count0 == 0) {
                    loop = 0; // 如果没有找到任何满足条件的位置，退出循环
                }
            }
            else
            {
                for (int i = 0; i < rows; i++)
                {
                    for (int j = 0; j < cols; j++)
                    {
                        draw_ball(i, j, arrayCopy0[i][j], 1);
                    }
                }
                int x0, y0;
                cout << endl;
                cct_getxy(x0, y0);
                cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
                for (int i = 0; i < rows; ++i)
                {
                    for (int j = 0; j < cols; ++j)
                    {
                        // 检查并标记可消除项
                        if (base_check2(arrayCopy0, directions, i, j, rows, cols))
                        {
                            draw_ball(i, j, arrayCopy0[i][j], 2);
                        }
                    }
                }
                cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
                cct_gotoxy(x0, y0);
            }
        }
        else
        {
            cct_gotoxy(x9, y9);
            cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
            cout << "不能交换" << char(i1 + 'A') << "行" << j1 + 1 << "列" << "<=>"
<< char(i2 + 'A') << "行" << j2 + 1 << "列";
            num0 = 0;
```

```
                                        }
                                    }
                                    num0++;
                                }
                                else
                                {
                                    cct_gotoxy(x9, y9);
                                    cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
                                    cout << "不能选择" << hang << "行" << lie << "列";
                                }
                                break;
                            case MOUSE_RIGHT_BUTTON_CLICK:
                                loop = 0;
                                cct_gotoxy(x9, y9);
                                break;
                        }
                    }

                    else
                    {
                        cct_gotoxy(x9, y9);
                        cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
                        cout << "位置非法";
                    }
                    cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
                    cout << "          ";
                    cct_gotoxy(x9, y9);

                }
                else
                {
                    cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
                    cct_gotoxy(x9, y9);
                    cout << "位置非法";
                    cout << "          ";
                    cct_gotoxy(x9, y9);
                }
            }
        }
        cct_setcolor(COLOR_BLACK, COLOR_WHITE);// 重置颜色
        cct_gotoxy(0, 0);
        cout << "                                                  ";
        cct_gotoxy(0, 0);
        cout << "无可消除项,游戏结束!最终分数:" << score << ")";
        cct_gotoxy(x9, y9);
        set_botton(0, 0);
}
/****************************************************************************
  函数名称: ball_base_all
  功    能: base的汇总函数
  输入参数:
  返 回 值:
  说    明:
****************************************************************************/
int ball_base_all(int choice)
{

                                        //这里的代码与9类似
            if (choice == 8)
            {
                int x8, y8;
                x8 = 0;
                y8 = 2 * rows + 2;
                cct_enable_mouse();
                cct_setcursor(CURSOR_INVISIBLE);
                while (loop)
                {
                    ret = cct_read_keyboard_and_mouse(X, Y, maction, keycode1, keycode2);

                    cout << "[当前光标] ";

                    if (ret == CCT_MOUSE_EVENT)
                    {
                        if (X <= 4 * cols && Y <= 2 * rows + 1)
                        {
                            hang = 'A' + Y / 2 - 1;
                            lie = int(X / 4) + 1;
                            if ((X % 4 == 2 || X % 4 == 3) && Y % 2 == 0 && Y > 0)
                            {
                                cout << hang << "行" << lie << "列";

                                switch (maction)
                                {
```

```
                                case MOUSE_LEFT_BUTTON_CLICK:
                                    if (base_check2(array, directions, X / 2 - 1, lie - 1, rows, cols))
                                    {
                                        cct_gotoxy(x8, y8);
                                        cout << "当前选择" << hang << "行" << lie << "列";
                                        Sleep(TIME * 20);

                                        loop = 0;
                                    }
                                    else
                                    {
                                        cct_gotoxy(x8, y8);
                                        cout << "不能选择" << hang << "行" << lie << "列";
                                    }
                                    break;
                                case MOUSE_RIGHT_BUTTON_CLICK:
                                    loop = 0;
                                    cct_gotoxy(x8, y8);
                                    break;
                                }
                            }

                            else
                            {
                                cout << "位置非法";
                            }
                            cout << "        ";
                            cct_gotoxy(x8, y8);

                        }
                        else
                        {
                            cout << "位置非法";
                            cout << "        ";
                            cct_gotoxy(x8, y8);
                        }
                    }
                }
            }
            cct_setcolor();
            set_botton(0, 0);
            break;
        case 9:
            draw_nine(array, rows, cols, choice);
            set_botton(0, 0);
            break;
    }
    if (choice != 7 && choice != 8 && choice != 9)
        cout << endl;
    }
    else
    {
        switch (choice)
        {
            case 8:
                                                    //这里与上面代码类似

}
return 0;
}
```