

## 高程magic\_ball大作业

答：上学期的夺冠其实是水到渠成，夺冠时其实是平静大于激动。然而下班学期随着高程的加入我们的训练变得越来越难了，人手不齐，甚至是上场的首发都不能全是主力，最后夺冠是喜出望外的。



朱俊泽

（高程磨练了我们球队的意志力，帮助我们济勤连夺两连冠！谢谢老师）

姓名：朱俊泽

学号：2351114

班级：信15

## 1. 题目

### 1.1. 要求

#### 1.1.1. 题目要求简述

题目第一项要求实现数组方式的初始化并且查找可消除项

题目第二项要求在第一项基础上实现消除，下落，新值填充的功能

题目第三项在第二项基础上在反复操作无初始可消除项之后显示可交换消除的提示项

题目第四项进行了伪图形初始化

题目第五项进行了高级的伪图形初始化

题目第六项在第四项基础上显示了初始可消除项

题目第七项在第五项基础始可消除项，消除，下落，新值填充，在反复进行之后无初始可消除项之后，显示可交换消除的提示项

题目第八项在第七项的基础上实现了鼠标操作选择可交换消除的提示项。

题目第九项就是本次magic\_ball大作业的最重要题目，实现了鼠标选择操作进行交换，消除，下落，填充，新值补充，分数计算还有判断如果没有可交换消除的提示项则游戏结束。

## 2. 整体设计思路

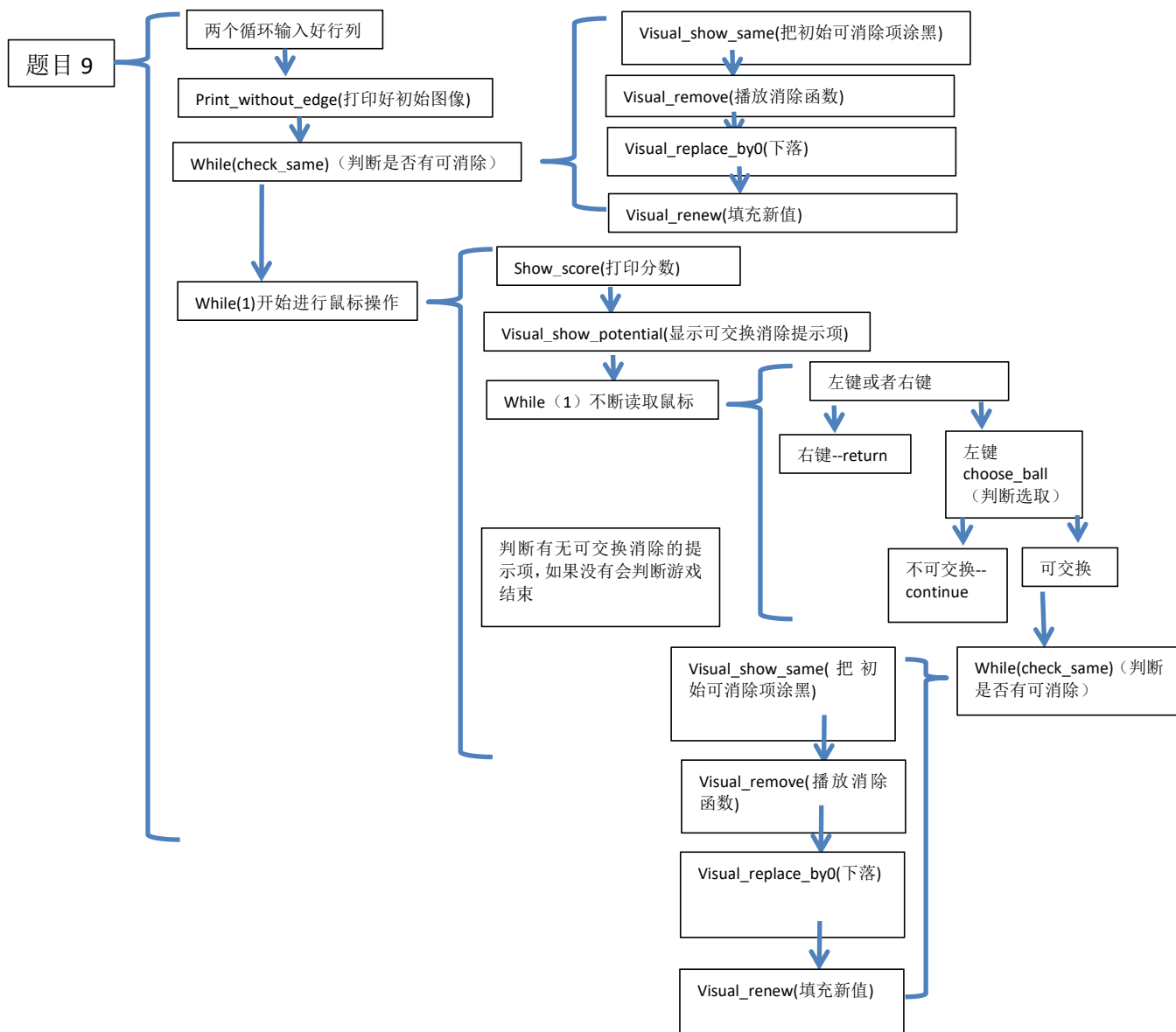
### 2.1. 部分非主要部分设计思路

这些部分包括了函数menu.cpp和主函数main.cpp。基本复用了汉诺塔的方式，就是多了一个输入end继续的部分，这里使用char数组就好了

### 2.2. 主要部分设计思路

主要部分只说明好题目第九项的要求就可以了，因为题目前三项的数组方式处理magic\_ball游戏的逻辑就是我在后面图形化后的内部数组处理逻辑，然后题目的4-7题都是图像化步骤，题目的第八项就是完成鼠标操作，都包含在了第九项之内。

第九项我的整体设计思路就是先进行内部数组的随机生成值初始化，再进行图像化生成，然后进入一个循环，这个循环的流程就是先判断目前有没有可以消除的项，有就进入循环，消除，下落，填充新值。直到跳出循环为止，然后此时我会判断有没有可以交换消除的提示项，没有的话我就会打印游戏结束，否则显示出可交换消除的提示项。此时进入一个读取鼠标操作的循环，这个循环的流程就是不断读取鼠标的位置，直到鼠标出现左键操作或者右键操作，如果是右键直接return表示游戏结束，不然就把此时的鼠标x，y坐标传出循环，判断位置的合法性以及选择结果。然后如果累计选择了两个相邻的并且两个交换后会形成初始可消除项的话，那就重新打印一遍整个图像，重新进入消除，下落，填充新值。此时需要记录消除了多少次，用来表示分数。这样就大致是主要部分的设计思路，下面附上一份图解帮助说明，对应函数在下方第3大题中有更加详细的说明。



## 3. 主要功能的实现

主要部分只说明第九项题目中使用到的函数就基本可以了，我解决本次大作业是顺序解决，题目前三项的数组方式处理magic\_ball游戏的逻辑就是我在后面图形化后的内部数组处理逻辑，然后题目的4-7题都是图像化步骤，题目的第八项就是完成鼠标操作，都包含在了第九项之内。

### 3.1. 非第九项的功能

#### 3.1.1. 菜单函数部分


正常清屏然后用循环提示输入0-9即可，控制输入正确范围再返回输入值。

## 3.1.2. 主函数部分

接受了菜单函数的返回值之后分情况进入game函数内设置好的情况,最后用char[]数组接受输入判断前三项是不是end就可以重新开始循环了

## 3.2. 第九项的功能函数(在整体设计思路中就设计到了--我按顺序介绍)


### 3.2.1. Print\_without\_edge

先用cct\_setfontsize改变好字体大小和边框,再对应设置好控制台的大小和缓冲区大小,然后在屏幕上方输出自己设置好的屏幕行和列。之后用输入的r行数,1列数来初始化图像。注意好附件中给的“”这些符号图像站两格宽一格高,也就是在输出的时候需要x+2,以2为单位计算。注意输出圆圈的时候颜色选择最好是mp【i】【j】+4,因为0-4中有几个颜色特别相似的蓝色,青色之类的。这样就能输出好图像框架。



### 3.2.2. check\_same

使用两个for循环判断有没有三连重复的即可,如果有就return true。如果跑完了两个循环判断行和列上都没有三重相同的,就return false。

### 3.2.3. Visual\_show\_same

和上方check\_same的逻辑相似,本函数中创建一个vis数组,初始化为0;用两个for循环判断三连相同的mp二维数组,如果相同就把vis设置为1,用一个for循环,如果vis【i】【j】为1就把对应位置打印成“”

### 3.2.4. Visual\_remove

和上方visual\_show\_same的逻辑相似,在判断后需要消除。消除的方式是循环五次,每一次在对应位置先打印,然后停顿一会再输出。最后打印成空白就代表消除了。同时把mp【i】【j】对应改成0,方便后面填充新值。

### 3.2.5. Visual\_replace\_by0

此前所有的遍历方式都是从1-》r,从1-》c。在这个下落函数中必须是从r-》1,从1-》c。我的操作方法是先从第一列最底部向上开始判断,如果mp【i】【j】是0,那么就向上找,直到找到一处mp【i+k】【j】不为0,就把他们两个数组互换,之后通过对坐标的控制打印将mp【i+k】【j】打印降落到mp【i】【j】位置,此后再向上判断,如此往复就可完成下落操作。

### 3.2.6. Visual\_renew

检查一遍整个mp数组,发现mp【i】【j】为0就随机一个新的数字填入并且打印就可以了

### 3.2.7. show\_score

这个接受了visual\_remove所return的值后就是新的score，对应位置输出即可

### 3.2.8. Visual\_show\_potential

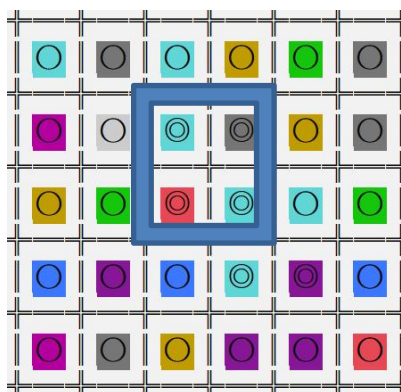
这个函数中我选择枚举判断每一处彩球在向四方交换后是否会出现可消除的情况，用上面提到的check\_same函数判断即可。开一个vis数组，初始化为0。向四个方向先交换两个彩球的颜色，然后check\_same，如果返回true，就把vis【i】【j】设置为1即可，在这样遍历完之后可以把vis数组遍历一遍，如果vis【i】【j】==1就对应输出 "◎"

### 3.2.9. Choose\_ball&game函数

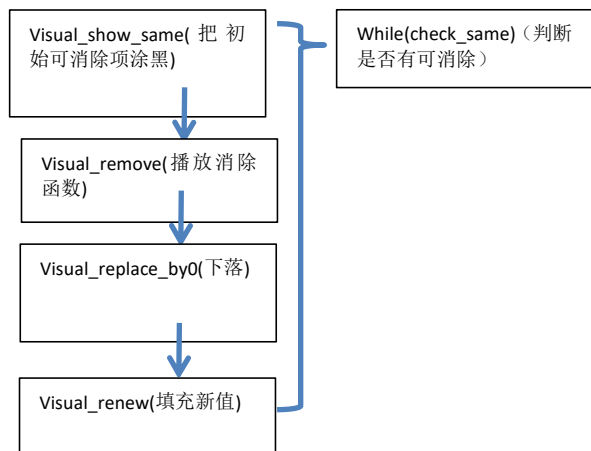
这个choose\_ball函数我希望减少传入的参数，因此我把选择和交换的判断放在了主要逻辑game函数里面，这里面我配合起来讲。先讲game函数：

从主函数main里面出来就是根据menu函数得到的选项进入game函数，前八项的逻辑基本包含在了第九项中，因此我主要讲述第九项的逻辑，在按照上述顺序进行到visual\_show\_potential函数的时候，已经显示出的可以交换进行消除的提示项。此时先把鼠标解禁cct\_enable\_mouse，再定义score得分为0；之后进入循环，不断用cct\_read\_keyboard\_and\_mouse函数读取鼠标的操作，相应判断位置是否合法，再读取鼠标操作，如果是右键就退出，如果是左键并且是合法的potential位置就进入以下循环：

choose\_ball来判断：如果是第一次选择，那就把目标位置打印成白色的"◎"就好，如果本次选择和上次选择一样就打印成黑色的"◎"表示取消本次选择，如果两次选择不是相邻的话，那就把上一次的输出成白色表示取消，把本次选择的输出成白色。如果两次是相邻的话，注意！！！此时需要判断一个问题，我在本次大作业中遇到的问题就是这个，要判断一下交换过后能不能使得mp二维数组出现使得check\_same返回true的值。原因如下



如上图中方框框内的四个potential显示的可交换消除的提示项，如果你把右上角的和右下角的交换，并不会产生三连相同的magic\_ball也就是无效交换，如果此时不判断就会出错。因此在判断后如果能够有效的交换那就在mp数组内把两者进行一次交换后进行一次如下循环，把其中的visual\_remove函数所返回的值加给score，让他在下一次初始可消除项消除结束后打印新的score分数。到此第九项基本逻辑就结束了。

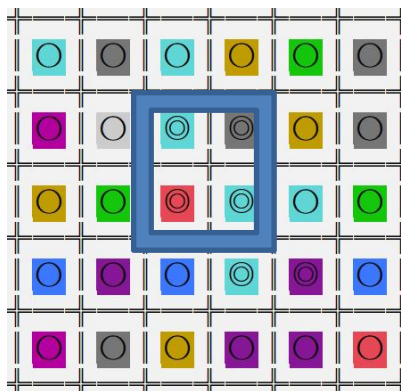


## 4. 调试过程碰到的问题

### 4.1. 问题一

#### 4.1.1. 问题描述

choose\_ball函数和game函数中部分内容负责判断选择可交换消除项的逻辑。在相邻的内容交换的时候，我发现了一个问题：



如上图中方框框内的四个potential显示的可交换消除的提示项，如果你把右上角的和右下角的交换，并不会产生三连相同的magic\_ball也就是无效交换，如果此时不判断就会出错。

#### 4.1.2. 问题解决

此时判断一下，具体判断方法：先交换mp【i】【j】和mp【i1】【j1】，然后判断check\_same（）如果返回false代表本次是无效交换，这个时候输出i行j列不能和i1行j1列交换，此时再交换回来即可。如果返回true代表本次其实是有效交换，此时就不用交换回来了，进入visual\_show\_same等循环开始执行消除，下落，新值填充等操作循环。

## 5. 心得体会

### 5.1. 心得体会，经验教训

本次大作业之中，首先写的三个小题就设计了后续第九题的内部数组处理逻辑，可是我把一二三题中数组处理和数组的输出写在了同一个函数，例如下图show\_same()函数

```
void show_same(int mp[12][12], int r, int c)
{
    int x, y;
    cct_getxy(x, y);
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c - 2; j++)
        {
            if (mp[i][j] == mp[i][j + 1] && mp[i][j + 1] == mp[i][j + 2])
            {
                for (int k = 0; k < 3; k++)
                {
                    cct_showch(2+3*(j+k), y - r + i - 1, mp[i][j]+'0', 14, 1, 1);
                }
            }
        }
    }

    for (int i = 1; i <= c; i++)
    {
        for (int j = 1; j <= r - 2; j++)
        {
            if (mp[j][i] == mp[j+1][i] && mp[j+1][i] == mp[j+2][i])
            {
                for (int k = 0; k < 3; k++)
                {
                    cct_showch(2 + 3 * i, y - r + j + k - 1, mp[j][i] + '0', 14, 1, 1);
                }
            }
        }
    }

    cct_gotoxy(x, y);
    cct_setcolor(0);
}
```

这样让这个函数复用性极差，在后续题目中根本无法调用，基本要把上半部分重新复制一遍，应该把数组处理和数组打印写成两个函数，这样在之后的第九题中就可以直接复用，提高代码的复用性。

其次还有一个问题，就是在整个程序的主题逻辑上，我基本是把 hanoi 的主题部分拿来复用了。用自上而下的思维编写的代码复用性会远远高于线性思维编写出的代码。因此写程序尤其是大项目最好是先规划好函数的功能再想着去如何实现，不然后面会经常为了前面的某一决定而多费力气。把前面的也改了也很费劲。

### 5.2. 通过综合题1/2中有关函数的分解与使用，总结你在完成过程中是否考虑了前后小题的关联关系，是否能尽可能做到后面小题有效利用前面小题已完成的代码，如何才能更好地重用代码？

前后两小题在主题的程序逻辑是一致的：main，menu函数，都是同一个框架，需要改变的就是game函数的内容。还有一个地方相同的是，两个综合题都是由数组模拟进化为图像化改造。

重用代码就要求我们对函数分解的足够细致又不会很繁琐。如果分解的过多，就会使得复用函数不如直接

写入代码；分解的过少，就会使得函数通用性过低，很难重用。就比如5-1中提到的前三小题的函数没有被第九题复用一样

```
void show_same(int mp[12][12], int r, int c)
{
    int x, y;
    cct_getxy(x, y);
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c - 2; j++)
        {
            if (mp[i][j] == mp[i][j + 1] && mp[i][j + 1] == mp[i][j + 2])
            {
                for (int k = 0; k < 3; k++)
                {
                    cct_showch(2 + 3 * (j + k), y - r + i - 1, mp[i][j] + '0', 14, 1, 1);
                }
            }
        }
    }

    for (int i = 1; i <= c; i++)
    {
        for (int j = 1; j <= r - 2; j++)
        {
            if (mp[j][i] == mp[j + 1][i] && mp[j + 1][i] == mp[j + 2][i])
            {
                for (int k = 0; k < 3; k++)
                {
                    cct_showch(2 + 3 * i, y - r + j + k - 1, mp[j][i] + '0', 14, 1, 1);
                }
            }
        }
    }

    cct_gotoxy(x, y);
    cct_setcolor(0);
}
```

这个函数如果分解成get\_same和show\_same的话，get用于获取有哪些可以消除的项；show用来cct\_showstr来打印。那么这样在项目9中就可以复用。

如果再有类似大作业，应该仔细分析每一题的要求，将他们分成比较细的部分，自上而下的考虑大作业问题，把各个小题目的相同项目画出，尽可能的规划为一个函数，把每个题目剩下的部分化为一整个函数。一定不要线性的解决问题。这样如果有错误居然需要逐字逐句的进行调试检查，非常的困难啊。

## 6. 附件：源程序

Print\_without\_edge



```

13 void print_without_edge(int mp[12][12], int r, int c, int op)
14 {
15     cct_setcolor(0);
16     cct_cls();
17     cct_setfontsize("新宋体", 36, 24);
18     if(op==4||op==6)
19         cct_setconsoleborder(40, (6+r), 40, (6+r));
20     else
21         cct_setconsoleborder(40, 5+r*2, 40, 5+r*2);
22     int row, column, rrow, rcolumn;
23     cct_getconsoleborder(row, column, rrow, rcolumn);
24     cout << "屏幕: " << column << "行" << row << "列";
25     if ((op == 6||op==7) && !check_same(mp, r, c))
26         cout << "(未找到初始可消除项)";
27     cout << endl;
28     int x, y;
29     cct_getxy(x, y);
30     for (int i = 1; i <= r; i++)
31     {
32         if (i == 1)
33         {
34             cct_showstr(x, y, "F", 15, 0, 1);
35             for (int j = 1; j <= c; j++)
36             {
37                 pause();
38                 if (op==4||op==6)
39                     cct_showstr(x+2*j, y, "=", 15, 0, 1);
40                 else
41                 {
42                     cct_showstr(x+4*j-2, y, "=", 15, 0, 1);
43                     if (j!=c)
44                         cct_showstr(x+4*j, y, "T", 15, 0, 1);
45                 }
46             }
47         }
48         if (op==4||op==6)
49             cct_showstr(x+2*(c)+2, y, "H", 15, 0, 1);
50         if (op==5||op==7)
51             cct_showstr(x+4*(c), y, "H", 15, 0, 1);
52     }
53     if (op == 4||op == 6)
54         cct_showstr(x, y+1, "I", 15, 0, 1);
55     else
56         cct_showstr(x, y+2*i-1, "I", 15, 0, 1);
57     for (int j = 1; j <= c; j++)
58     {
59         if (op==4||op==6)
60             cct_showstr(x+2*j, y+i, "O", mp[i][j]+4, 0, 1);
61         else
62         {
63             cct_showstr(x+4*j-2, y+2*i-1, "O", mp[i][j]+4, 0, 1);
64             cct_showstr(x+4*j, y+2*i-1, "I", 15, 0, 1);
65         }
66     }
67     pause();
68     if (op==4||op==6)
69         cct_showstr(x+2*(c+1), y+i, "I", 15, 0, 1);
70     if (i == r && (op == 4||op == 6))
71     {
72         cct_showstr(x, y+i+1, "L", 15, 0, 1);
73         for (int j = 1; j <= c; j++)
74         {
75             pause();
76             cct_showstr(x+2*j, y+i+1, "=", 15, 0, 1);
77         }
78         cct_showstr(x+2*(c+1), y+i+1, "d", 15, 0, 1);
79     }
80     else if (i==r && (op == 5||op == 7))

```

```

80     else if (i==r && (op == 5||op == 7))
81     {
82         cct_showstr(x, y+2*r, "L", 15, 0, 1);
83         for (int j = 1; j <= c; j++)
84         {
85             pause();
86             cct_showstr(x+4*j-2, y+2*r, "=", 15, 0, 1);
87             cct_showstr(x+4*j, y+2*r, "d", 15, 0, 1);
88         }
89         cct_showstr(x+4*(c), y+2*r, "d", 15, 0, 1);
90     }
91     if (i != r && (op == 5||op==7))
92     {
93         cct_showstr(x, y+2*i+1, "I", 15, 0, 1);
94         for (int j = 1; j <= c; j++)
95         {
96             pause();
97             cct_showstr(x+4*j-2, y+2*i+1, "=", 15, 0, 1);
98             cct_showstr(x+4*j, y+2*i+1, "d", 15, 0, 1);
99         }
100         cct_showstr(x+4*(c), y+2*i+1, "d", 15, 0, 1);
101     }
102     cct_setcolor(0);
103 }
104

```

Check\_same

Visual\_show\_same

```

bool check_same(int mp[12][12], int r, int c)
{
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c-2; j++)
        {
            if (mp[i][j] == mp[i][j+1] && mp[i][j+1] == mp[i][j+2])
            {
                return true;
            }
        }
    }
    for (int i = 1; i <= c; i++)
    {
        for (int j = 1; j <= r-2; j++)
        {
            if (mp[j][i] == mp[j+1][i] && mp[j+1][i] == mp[j+2][i])
            {
                return true;
            }
        }
    }
    return false;
}

void visual_show_same(int mp[12][12], int r, int c, int op)
{
    int vis[12][12];
    memset(vis, 0, sizeof vis);
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c-2; j++)
        {
            if (mp[i][j] == mp[i][j+1] && mp[i][j+1] == mp[i][j+2])
            {
                for (int k = 0; k < 3; k++)
                {
                    vis[i][j+k] = 1;
                }
            }
        }
    }
    for (int i = 1; i <= c; i++)
    {
        for (int j = 1; j <= r-2; j++)
        {
            if (mp[j][i] == mp[j+1][i] && mp[j+1][i] == mp[j+2][i])
            {
                for (int k = 0; k < 3; k++)
                {
                    vis[j+k][i] = 1;
                }
            }
        }
    }
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c; j++)
        {
            if (op == 6)
            {
                if (vis[i][j])
                    cct_showstr(2*j-1, i+1, "●", mp[i][j]+4, 0, 1);
            }
            if (op == 7)
            {
                if (vis[i][j])
                    cct_showstr(4*j-2, i+1, "●", mp[i][j]+4, 0, 1);
            }
        }
    }
    cct_setcolor(0);
}

```

Visual\_remove

Visual\_replace\_by0

```
int visual_remove(int mp[12][12], int r, int c, int op)
{
    int res = 0;
    int vis[12][12];
    memset(vis, 0, sizeof vis);
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c - 2; j++)
        {
            if (mp[i][j] == mp[i][j + 1] && mp[i][j + 1] == mp[i][j + 2])
            {
                for (int k = 0; k < 3; k++)
                {
                    vis[i][j + k] = 1;
                }
            }
        }
    }

    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c - 2; j++)
        {
            if (mp[j][i] == mp[j + 1][i] && mp[j + 1][i] == mp[j + 2][i])
            {
                for (int k = 0; k < 3; k++)
                {
                    vis[j + k][i] = 1;
                }
            }
        }
    }

    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c; j++)
        {
            if (vis[i][j])
            {
                res++;
            }
        }
    }

    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c; j++)
        {
            if (vis[i][j])
            {
                pause(500);
                dfs(mp, vis, r, c, op, i, j);
            }
        }
    }

    cct_setcolor(0);
    return res;
}

void visual_replace_by_0(int mp[12][12], int r, int c)
{
    for (int j = 1; j <= c; j++)
    {
        for (int i = r; i >= 1; i--)
        {
            if (mp[i][j] == 0)
            {
                for (int k = i; k >= 1; k--)
                {
                    int cnt = 1;
                    if (mp[k][j] != 0)
                    {
                        while (mp[k + cnt][j] == 0 && (k + cnt <= r))
                        {
                            cnt++;
                        }
                        cnt--;
                        for (int l = 1; l <= cnt; l++)
                        {
                            pause(50);
                            cct_showstr(4 * j - 2, 2 * k + (l - 1) * 2, 'O', 15, 15, 1);
                            pause(50);
                            cct_showstr(4 * j - 2, 2 * k + (l - 1) * 2, 'O', mp[k][j] + 4, 0, 1);
                            pause(50);
                            cct_showstr(4 * j - 2, 2 * k + (l - 1) * 2, '-', 15, 0, 1);
                            pause(50);
                            cct_showstr(4 * j - 2, 2 * k + (l - 1) * 2, 'O', mp[k][j] + 4, 0, 1);
                        }
                        mp[k + cnt][j] = mp[k][j];
                        mp[k][j] = 0;
                    }
                }
            }
        }
    }

    cct_setcolor(0);
}
```

Visual\_renew

show\_score

```
void visual_renew(int mp[12][12], int r, int c)
{
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c; j++)
        {
            if (mp[i][j] == 0)
            {
                pause(100);
                mp[i][j] = rand() % 9 + 1;
                cct_showstr(4 * j - 2, 2 * i, 'O', mp[i][j] + 4, 0, 1);
            }
        }
    }

    cct_setcolor(0);
}

void show_score(int now)
{
    cct_gotoxy(14, 0);
    cout << " (当前分数:";
    cout << std::right << setw(3) << now;
    cout << " 右键退出) ";
}
```

Visual\_show\_potential

```
void visual_show_potential(int vis[12][12], int mp[12][12], int r, int c)
{
    int dx[5] = {0, 0, 1, -1};
    int dy[5] = {1, -1, 0, 0};
    memset(vis, 0, sizeof vis);
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c; j++)
        {
            for (int k = 0; k < 4; k++)
            {
                int nx = i + dx[k], ny = j + dy[k];
                int temp = mp[i][j];
                mp[i][j] = mp[nx][ny];
                mp[nx][ny] = temp;
                if (check_same(mp, r, c))
                {
                    for (int ni = 1; ni <= r; ni++)
                    {
                        for (int nj = 1; nj <= c - 2; nj++)
                        {
                            if (mp[ni][nj] == mp[ni][nj + 1] && mp[ni][nj + 1] == mp[ni][nj + 2])
                            {
                                vis[i][j] = 1;
                                vis[nx][ny] = 1;
                            }
                        }
                    }
                }
            }
        }
    }

    for (int ni = 1; ni <= c; ni++)
    {
        for (int nj = 1; nj <= r - 2; nj++)
        {
            if (mp[nj][ni] == mp[nj + 1][ni] && mp[nj + 1][ni] == mp[nj + 2][ni])
            {
                vis[i][j] = 1;
                vis[nx][ny] = 1;
            }
        }
    }

    mp[nx][ny] = mp[i][j];
    mp[i][j] = temp;

    pause(100);
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c; j++)
        {
            if (vis[i][j])
            {
                cct_showstr(4 * j - 2, 2 * i, "O", mp[i][j] + 4, 0, 1);
            }
            else
            {
                cct_showstr(4 * j - 2, 2 * i, "O", mp[i][j] + 4, 0, 1);
            }
        }
    }

    cct_setcolor(0);
}
```

Game

```

void game(int op)
{
    int mp[12][12];
    memset(mp, 0, sizeof mp);
    int r, c;
    cct_cls();
    while (1) { ... }
    while (1) { ... }
    if (op == 1 || op == 2 || op == 3) { ... }
    if (op == 4 || op == 5 || op == 6 || op == 7) { ... }
    if (op == 8) { ... }
    if (op == 9) { ... }
}

```

```

if (op == 9)
{
    int vis[12][12];
    memset(vis, 0, sizeof vis);
    for (int i = 1; i <= r; i++)
    {
        for (int j = 1; j <= c; j++)
        {
            initial_mp(mp, i, j);
        }
    }
    print_without_edge(mp, r, c, 7);
    int x, y;
    cct_getxy(x, y);
    while (check_same(mp, r, c))
    {
        visual_show_same(mp, r, c, 7);
        pause(100);
        visual_remove(mp, r, c, op);
        pause(100);
        visual_replace_by_0(mp, r, c);
        pause(100);
        visual_renew(mp, r, c);
    }
    visual_show_potential(vis, mp, r, c);
    cct_enable_mouse();
    int score = 0;
    int pr = 0, pc = 0;
    while (1)
    {

```

```

        pc = 0;
        else
        {
            pr = (m_y) / 2;
            pc = (m_x + 1) / 2;
        }
        else
        {
            cct_gotoxy(x, y);
            cout << endl << " ";
            cct_gotoxy(x, y);
            cout << endl << "不能选择" << char_r << "行" << int_c << "列";
            pause(200);
        }
    }
    cct_gotoxy(x, y);
}

```

```

while (1)
{
    //visual_show_potential(vis, mp, r, c);
    if (check_end(vis, r, c))
        break;
    cct_setcolor(0);
    show_score(score);
    int m_x = 0, m_y = 0, m_action = 0, k1 = 0, k2 = 0;
    int res = cct_read_keyboard_and_mouse(m_x, m_y, m_action, k1, k2);
    m_x /= 2;
    char char_r = 'A' + (m_y - 2) / 2;
    int int_c = (m_x + 1) / 2;
    cct_gotoxy(x, y);
    if ((m_y % 2 == 0) && (m_x % 2 == 1) && ((m_y - 2) / 2 < r) && ((m_x + 1) / 2 <= c))
        cout << endl << "[当前光标]" << char_r << "行" << int_c << "列";
    else
        cout << endl << "[当前光标]" << "位置非法";
    if (m_action == MOUSE_RIGHT_BUTTON_CLICK)
        return;
    if (m_action == MOUSE_LEFT_BUTTON_CLICK)
    {
        if ((m_y % 2 == 0) && (m_x % 2 == 1))
        {
            visual_show_potential(vis, mp, r, c);
            if (vis[(m_y) / 2][(m_x + 1) / 2])
            {
                cct_gotoxy(x, y);
                cout << endl << " ";
                cct_gotoxy(x, y);
                cout << endl << "当前选择" << char_r << "行" << int_c << "列";
                pause(200);
            }

```

```

        cct_gotoxy(x, y);
        cout << endl << "当前选择" << char_r << "行" << int_c << "列";
        pause(200);
        bool check_choose_ball(mp, (m_y) / 2, (m_x + 1) / 2, pr, pc);
        if (check_choose_ball(mp, r, c) && check)
        {
            while (check_choose_ball(mp, r, c))
            {
                cct_showstr(4 * pc - 2, 2 * pr, "O", mp[pr][pc] + 4, 0, 1);
                cct_showstr(4 * (m_x + 1) / 2 - 2, 2 * (m_y) / 2, "O", mp[(m_y) / 2][(m_x + 1) / 2] + 4, 0, 1);
                visual_show_same(mp, r, c, 7);
                pause(100);
                score += visual_remove(mp, r, c, op);
                pause(100);
                visual_replace_by_0(mp, r, c);
                pause(100);
                visual_renew(mp, r, c);
                memset(vis, 0, sizeof vis);
                visual_show_potential(vis, mp, r, c);
            }
            pr = 0;
            pc = 0;
        }
        else if (check)
        {
            cct_setcolor(0);
            cct_gotoxy(x, y);
            cout << endl << " ";
            cct_gotoxy(x, y);
            cout << endl << "不能交换" << 'A' + pr - 1 << "行" << pc << "列" << "<" << char_r << "行" << int_c << "列";
            pause(200);
            int temp = mp[(m_y) / 2][(m_x + 1) / 2];
            mp[(m_y) / 2][(m_x + 1) / 2] = mp[pr][pc];
            mp[pr][pc] = temp;
            cct_showstr(4 * pc - 2, 2 * pr, "O", mp[pr][pc] + 4, 0, 1);
            cct_showstr(4 * (m_x + 1) / 2 - 2, 2 * (m_y) / 2, "O", mp[(m_y) / 2][(m_x + 1) / 2] + 4, 0, 1);
            pr = 0;
            pc = 0;
        }
        else

```

Choose\_ball

```

345 bool choose_ball(int mp[12][12], int nr, int nc, int pr, int pc)
346 {
347     if (pr == 0 && pc == 0)
348     {
349         cct_showstr(4 * nc - 2, 2 * nr, "O", mp[nr][nc] + 4, 15, 1);
350     }
351     else if (nr == pr && nc == pc)
352     {
353         cct_showstr(4 * nc - 2, 2 * nr, "O", mp[nr][nc] + 4, 0, 1);
354     }
355     else if ((abs(nr - pr) + abs(nc - pc)) > 1)
356     {
357         cct_showstr(4 * nc - 2, 2 * pr, "O", mp[pr][nc] + 4, 0, 1);
358         cct_showstr(4 * nc - 2, 2 * nr, "O", mp[nr][nc] + 4, 15, 1);
359     }
360     else
361     {
362         int temp = mp[nr][nc];
363         mp[nr][nc] = mp[pr][pc];
364         mp[pr][pc] = temp;
365         return true;
366     }
367     cct_setcolor(0);
368     return false;
369 }

```