第7讲:数组和函数实践:扫雷游戏

目录

- 1. 扫雷游戏分析和设计
- 2. 扫雷游戏的代码实现
- 3. 扫雷游戏的扩展

正文开始

1. 扫雷游戏分析和设计

1.1 扫雷游戏的功能说明

- 使用控制台实现经典的扫雷游戏
- 游戏可以通过菜单实现继续玩或者退出游戏
- 扫雷的棋盘是9*9的格子
- 默认随机布置10个雷
- 可以排查雷
 - 如果位置不是雷,就显示周围有几个雷
 - 。 如果位置是雷,就炸死游戏结束
 - 。 把除10个雷之外的所有雷都找出来,排雷成功,游戏结束

游戏的界面:

***** 1.								play				* **	**
**	* *	* **	* *		().	ez	xi1	t		**	* **	**
**	< * >	* *	* *>	<u></u> **	* *>	* *>	<u></u> **	* *>	***	* **	**	* **	**
谴	拼	括	<u>:</u> :)	>1									
**	<*>	k*>	* *	#	1雷	; ;	* *>	* *>	* **				
0	1	2	3	4	5	6	7	8	9				
1	*	*	*	*	*	*	*	*	*				
2	*	*	*	*	*	*	*	*	*				
3	*	*	*	*	*	*	*	*	*				
4	*	*	*	*	*	*	*	*	*				
5	*	*	*	*	*	*	*	*	*				
6	*	*	*	*	*	*	*	*	*				
7	*	*	*	*	*	*	*	*	*				
8	*	*	*	*	*	*	*	*	*				
9	*	*	*	*	*	*	*	*	*				
٧.		įλ		· 扫	· 查		· J坐	· 公标	: :>				

比特就	i 🏰	隅	原页	、溥	TH.	S 古	帲	194	4新	k.com/	inst/	s9ye	whfr
	*>	**	* *>	* *	`担	霍			_ , ,	k**	Ŭ		
	0	1	2	3	4	5	6	7	8	9			
	1	0	0	0	0	0	1	*	*	*			
	2	0	0	0	0	0	1	*	*	*			
	3	*	*	*	*	*	*	*	*	*			
	4	*	*	*	*	*	*	*	*	*			
	5	*	*	*	*	*	*	*	*	*			
	6	*	*	*	*	*	*	*	*	*			
	7	*	*	*	*	*	*	*	*	*			
	8	*	*	*	*	*	*	*	*	*			
	9	*	*	*	*	*	*	*	*	*			
	禕	育箱	ijλ	要	抖	查	的	了坐	经标	₹:>			

排雷界面



排雷失败界面

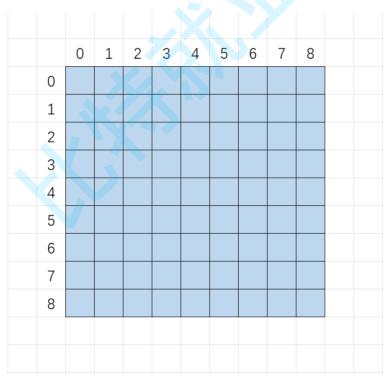
初始界面

1.2 游戏的分析和设计

1.2.1 数据结构的分析

扫雷的过程中,布置的雷和排查出的雷的信息都需要存储,所以我们需要一定的数据结构来存储这些信息。

因为我们需要在9*9的棋盘上布置雷的信息和排查雷,我们首先想到的就是创建一个9*9的数组来存放信息。



空棋盘

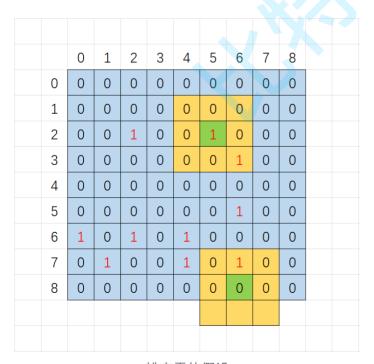
那如果这个位置布置雷,我们就存放1,没有布置雷就存放0.



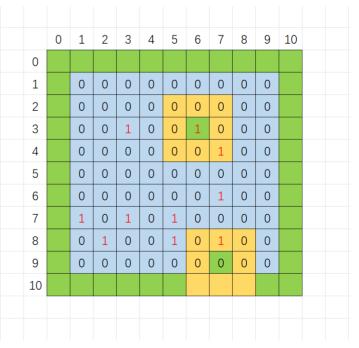
布置雷的棋盘

假设我们排查(2,5)这个坐标时,我们访问周围的一圈8个黄色位置,统计周围雷的个数是1

假设我们排查(8,6)这个坐标时,我们访问周围的一圈8个黄色位置,统计周围雷的个数时,最下面的三个坐标就会越界,为了防止越界,我们在设计的时候,给数组扩大一圈,雷还是布置在中间的9*9的坐标上,周围一圈不去布置雷就行,这样就解决了越界的问题。所以我们将存放数据的数组创建成11*11是比较合适。



排查雷的假设



雷区外多加一圈防止越界

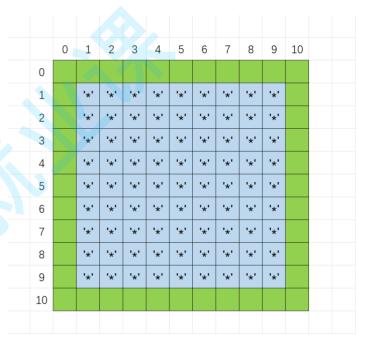
再继续分析,我们在棋盘上布置步蘭,積盈上當的信息¹(幻)²和非智的信息(0),假设我们排查了某一个位置后,这个坐标处不是雷,这个坐标的周围有1个雷,那我们需要将排查出的雷的数量信息记录存储,并打印出来,作为排雷的重要参考信息的。那这个雷的个数信息存放在哪里呢?如果存放在布置雷的数组中,这样雷的信息和雷的个数信息就可能或产生混淆和打印上的困难。

这里我们肯定有办法解决,比如:雷和非雷的信息不要使用数字,使用某些字符就行,这样就避免冲 突了,但是这样做棋盘上有雷和非雷的信息,还有排查出的雷的个数信息,就比较混杂,不够方便。

这里我们采用另外一种方案,我们专门给一个棋盘(对应一个数组mine)存放布置好的雷的信息,再给另外一个棋盘(对应另外一个数组show)存放排查出的雷的信息。这样就互不干扰了,把雷布置到mine数组,在mine数组中排查雷,排查出的数据存放在show数组,并且打印show数组的信息给后期排查参考。

同时为了保持神秘,show数组开始时初始化为字符 '*',为了保持两个数组的类型一致,可以使用同一套函数处理,mine数组最开始也初始化为字符'0',布置雷改成'1'。如下如:





mine数组布置雷后的状态

show数组初始状态

对应的数组应该是:

```
1 char mine[11][11] = {0};//用来存放布置好的雷的信息
2 char show[11][11] = {0};//用来存放排查出的雷的个数信息
```

1.2.2 文件结构设计

之前学习了多文件的形式对函数的声明和定义,这里我们实践一下,我们设计三个文件:

- 1 test.c //文件中写游戏的测试逻辑
- 2 game.c //文件中写游戏中函数的实现等
- 3 game.h //文件中写游戏需要的数据类型和函数声明等

比特就业课-专注IT大学生就业的精品课程

2. 扫雷游戏的代码实现

game.h

```
1 #pragma once
 2
 3 #include <stdio.h>
 4 #include <stdlib.h>
 5 #include <time.h>
7 #define EASY_COUNT 10
8
9 #define ROW 9
10 #define COL 9
11
12 #define ROWS ROW+2
13 #define COLS COL+2
14
15 //初始化棋盘
16 void InitBoard(char board[ROWS][COLS], int rows, int cols, char set);
17
18 //打印棋盘
19 void DisplayBoard(char board[ROWS][COLS], int row, int col);
20
21 //布置雷
22 void SetMine(char board[ROWS][COLS], int row, int col);
23
24 //排查雷
25 void FindMine(char mine[ROWS][COLS], char show[ROWS][COLS], int row, int col);
26
```

game.c

```
1 #include "game.h"
2
3 void InitBoard(char board[ROWS][COLS], int rows, int cols, char set)
4 {
5     int i = 0;
6     for (i = 0; i < rows; i++)
7     {
8        int j = 0;
        Lt特就业课-专注IT大学生就业的精品课程</pre>
```

```
for (片特就业课主页: https://m.cctalk.com/inst/s9yewhfr
9
                    {
10
                            board[i][j] = set;
11
12
                    }
           }
13
14 }
15
16 void DisplayBoard(char board[ROWS][COLS], int row, int col)
17
18
           int i = 0;
           printf("------扫雷游戏-----\n");
19
           for (i = 0; i <= col; i++)
20
21
22
                    printf("%d ", i);
           }
23
           printf("\n");
24
           for (i = 1; i <= row; i++)
25
26
           {
27
                    printf("%d ", i);
28
                    int j = 0;
                    for (j = 1; j \le col; j++)
29
30
                            printf("%c ", board[i][j]);
31
32
                    }
                    printf("\n");
33
           }
34
35 }
36
37 void SetMine(char board[ROWS][COLS], int row, int col)
38 {
           //布置10个雷
39
           //生成随机的坐标,布置雷
40
           int count = EASY_COUNT;
41
42
           while (count)
43
           {
44
                    int x = rand() % row + 1;
                    int y = rand() % col + 1;
45
46
                    if (board[x][y] == '0')
47
                    {
48
                            board[x][y] = '1';
49
50
                            count--;
                    }
51
52
           }
53 }
54
55 int GetMineCount(char mine[ROWS][COLS], int x, int y)
                             比特就业课-专注IT大学生就业的精品课程
```

```
比特就业课主页:https://m.cctalk.com/inst/s9yewhfr
56 {
57
           return (mine[x-1][y]+mine[x-1][y-1]+mine[x][y-1]+mine[x+1][y-1]+mine[x
                   mine[x+1][y+1]+mine[x][y+1]+mine[x-1][y+1] - 8 * '0');
58
59 }
60
61
62 void FindMine(char mine[ROWS][COLS], char show[ROWS][COLS], int row, int col)
63 {
64
           int x = 0;
           int y = 0;
65
66
           int win = 0;
           while (win <row*col- EASY_COUNT)</pre>
67
68
                   printf("请输入要排查的坐标:>");
69
                   scanf("%d %d", &x, &y);
70
                   if (x \ge 1 \&\& x \le row \&\& y \ge 1 \&\& y \le col)
71
72
                   {
73
                           if (mine[x][y] == '1')
74
                            {
                                    printf("很遗憾,你被炸死了\n");
75
76
                                    DisplayBoard(mine, ROW, COL);
                                    break;
77
                            }
78
79
                            else
                            {
80
                                    //该位置不是雷,就统计这个坐标周围有几个雷
81
82
                                    int count = GetMineCount(mine, x, y);
83
                                    show[x][y] = count + '0';
                                    DisplayBoard(show, ROW, COL);
84
                                    win++;
85
86
                   }
87
                   else
88
                   {
89
90
                           printf("坐标非法,重新输入\n");
91
                   }
92
           }
           if (win == row * col - EASY_COUNT)
93
           {
94
                   printf("恭喜你,排雷成功\n");
95
                   DisplayBoard(mine, ROW, COL);
96
           }
97
98 }
```

test.c

```
比特就业课主页:https://m.cctalk.com/inst/s9yewhfr
1 #include "game.h"
2
3 void menu()
4 {
           printf("*************************);
5
           printf("***** 1. play
6
                                   ****\n");
           printf("*****
7
                         0. exit
                                    ****\n");
           printf("************************);
8
9 }
10
11 void game()
12 {
           char mine[ROWS][COLS];//存放布置好的雷
13
           char show[ROWS][COLS];//存放排查出的雷的信息
14
           //初始化棋盘
15
           //1. mine数组最开始是全'0'
16
           //2. show数组最开始是全'*'
17
18
           InitBoard(mine, ROWS, COLS, '0');
19
           InitBoard(show, ROWS, COLS, '*');
           //打印棋盘
20
21
           //DisplayBoard(mine, ROW, COL);
           DisplayBoard(show, ROW, COL);
22
           //1. 布置雷
23
           SetMine(mine, ROW, COL);
24
           //DisplayBoard(mine, ROW, COL);
25
           //2. 排查雷
26
           FindMine(mine, show, ROW, COL);
27
28 }
29
30 int main()
31 {
           int input = 0;
32
           srand((unsigned int)time(NULL));
33
           do
34
35
           {
36
                   menu();
                   printf("请选择:>");
37
                   scanf("%d", &input);
38
                   switch (input)
39
                   {
40
                   case 1:
41
42
                           game();
43
                           break;
44
                   case 0:
                           printf("退出游戏\n");
45
46
                           break;
47
                   default:
                            比特就业课-专注IT大学生就业的精品课程
```

3. 扫雷游戏的扩展

- 是否可以选择游戏难度
 - 简单 9*9 棋盘, 10个雷
 - 中等 16*16棋盘,40个雷
 - 困难 30*16棋盘,99个雷
- 如果排查位置不是雷,周围也没有雷,可以展开周围的一片
- 是否可以标记雷
- 是否可以加上排雷的时间显示

在线扫雷游戏: https://www.msn.cn/zh-cn/play/microsoft-minesweeper/cg-msminesweeper

完