1. 扫雷游戏

目的：创建一个9\*9的扫雷游戏

着重考虑：棋盘的初始化，放置雷，扫除雷，输入一个坐标，旁边没雷的地方变为空地

详细步骤：

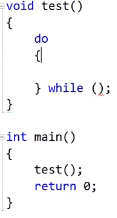
需要测试这个游戏，需要一个test.c文件

还需要实现游戏的文件game.c文件，同时需要申明函数的game.h头文件



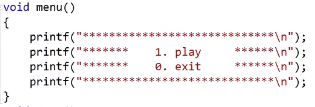
测试文件（test.c）里写入mine函数用来测试代码

封装test函数，为了可以一直玩游戏玩下去，写入do while结构



游戏进入需要打印一份菜单

设置一个menu函数供玩家选择，不需要返回值设置成void类型



test函数里放置菜单函数，设置scanf函数让玩家输入数字供选择菜单项目

如果是1就进入游戏，0就退出游戏，如果非1非0 进入default让玩家重新选择



while后括号加input，如果是非零则保持循环，输入零的时候打印“退出游戏”，跳出循环

在case1中写入game函数

//雷的信息储存

定义game 函数，不需要返回值所以用void

在game 函数中创建数组 char board[9][9]

如果放雷，雷是1，那么其他的地方就是0



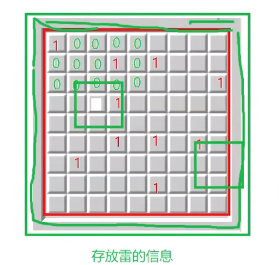
数组一开始都是0，布置雷的时候放1进去，为了保持神秘，用\*覆盖未排查的坐标

假设用户输入一个坐标，如果坐标不是雷，就输出“游戏结束”

如果不是雷，那么就应该显示周围有多少雷

但是如果坐标显示的是1，那么对于这个坐标本身就是雷还是周围有一个雷就产生了歧义

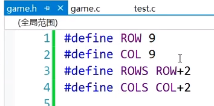
游戏目前需要两个数组，一个存放雷的信息（程序员后台可见），一个存放排查出来的雷的信息（供玩家可见，未选中的坐标用\*代替）

在排查雷的时候，如果是排查边边的坐标，很容易造成越界

最好的办法是将棋盘扩大一圈，9\*9的棋盘变成11\*11空出来的一圈全放0

在game函数里写入布置好雷的数组

char mine[ROWS][COLS]={0}；(为了方便修改，横纵定义在game.h中，11\*11也顺便定义了)



既然使用了game.h文件里的东西，就要引用头文件（#include”game.h”）

再设置展示出去的数组，放排查出雷的信息

char show[ROWS][COLS]={0}；

定义一个初始化函数InitBoard()

{

char mine[ROWS][COLS]={0}；（没有排查的时候希望布置字符0）

char show[ROWS][COLS]={0}；（未排查雷的时候希望放置字符\*）

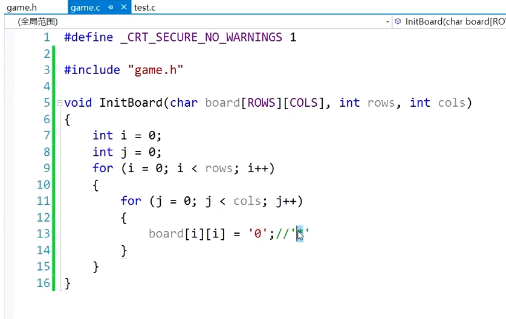
}

InitBoard(mine，ROWS,COLS):想先初始化mine函数先将mine传过去，初始化ROWS这么多列，COLS这么多行）

这个函数想要实现的话需要先进行申明，在game.h中进行申明

InitBoard(char board[ROWS][COLS],int rows,int cols)

申明过之后，就在game.c中使用



因为用了game.h文件中的定义，所以依然需要引用头文件

但是这个函数只能把数组初始化成一样的

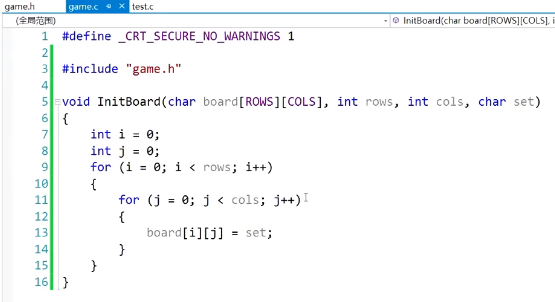
不如把要初始化的内容也传过去



申明函数也随之改变



使用函数也改变



调的时候给字符0就传字符0，给\*就传\*

为了放置后续错误堆积，不如先打印出来看看

写一个DisplayBoard（）函数用于打印，传参



相同的在头文件中申明

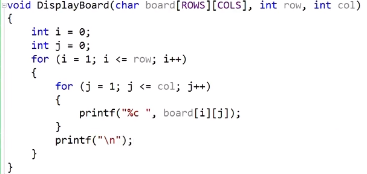


因为mine中传的参是[ROWS][COLS]，所以申明的是char board[ROWS][COLS]

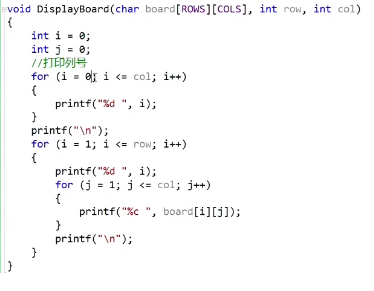
申明好之后在game.c中使用

打印棋盘无非就是把数组的内容打印出来

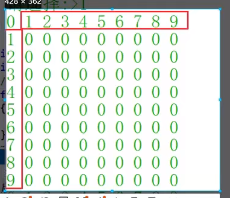
存放雷的坐标行和列都从1开始，所以for循环中，i和j分别<=row，col



但是为了方便用户选择坐标，我们可以把行号，列号都加上



打印效果：



棋盘打印出来以后，接下来就应该布置雷

布置雷其实就是把雷的信息放到mine数组里面去，在mine数组里面放上字符1

而且是随机的坐标。范围是9\*9的方格

设置函数SetMine(mine,ROW,COL)//布置雷

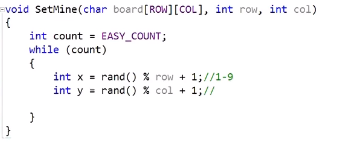
同样需要申明在头文件中

void SetMine(char board[ROWS][COLS], int row , int col );

接下来可以实现了

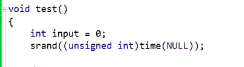
需要定义放入几个雷，为了方便以后修改，在头文件中定义一个EASY\_COUNT 为10

布置雷就是在棋盘上找一个随机的坐标，如果里面没有雷就可以布置雷进去，且范围在9\*9的方格内，也就是说行和列的范围都在1-9

先生成随机的坐标

rand的返回值是介于0和[RAND\_MAX](http://www.cplusplus.com/RAND_MAX)之间的整数值。且任何数除以9都是在0-8之间，再+1范围就是1-9之间

生成随机的坐标之后，rand这个函数在使用之前必须要调用一个函数srand(只要调用一次,所以放在test内部)

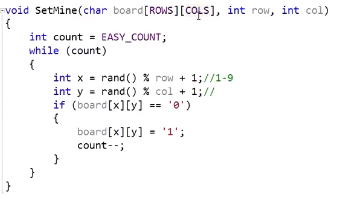


用time函数的返回值作为随机数的生成起点，同时把返回类型强制类型转换，转换成unsigned int ，调srand 时，time函数的返回值生成时间戳设置srand

调用函数同样需要在头文件中引用

如果，一个坐标的值是字符‘0’，那么其中可以布置雷，放入字符‘1’，布置成功之后，count--

直到count为0时停止循环



布置雷的效果图



传参的时候传的是什么样的数组形参就是什么数组



数组传参的时候虽然是11\*11但是操作的范围依然是9\*9

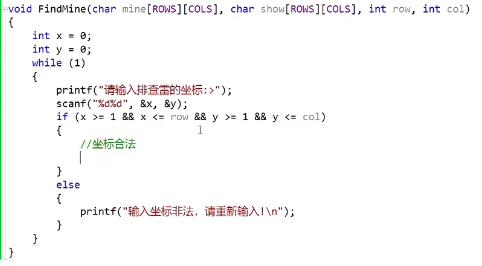
接下来是扫雷的过程,在mine数组里面找信息，找到之后传到show数组里面去

操作范围是9\*9

写入函数FindMine(mine,show,ROW ,COL);

申明void FindMine（char mine[ROWS][COLS],char show[ROWS][COLS],int row ,int col）;

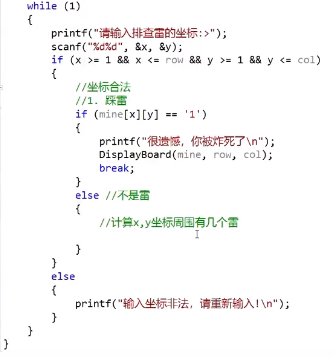
实现



当坐标合法的时候，应该先判断当前坐标是不是雷，再统计周围有几个雷

如果是雷，告知用户被炸死了，并且将雷打印出来，游戏终止break跳出

将for循环改成while循环

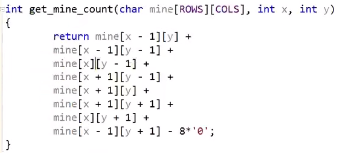


如果坐标不是雷，计算（x,y）坐标周围雷的数量（封装函数get\_mine\_count(mine,x,y)统计雷的个数）

字符1减去字符0得到的就是数字1

字符3减去字符0得到的就是数字3





设置拓展函数，如果一个坐标附近大面积没有雷的话就展开，提高游戏效率

