第7讲：扫雷游戏（基础实现）

1、扫雷游戏分析和设计

1.1 游戏要求

（1）使用控制台实现经典的扫雷游戏

（2）游戏可以通过菜单实现继续玩或者退出游戏

（3）扫雷的棋盘是9\*9的格子

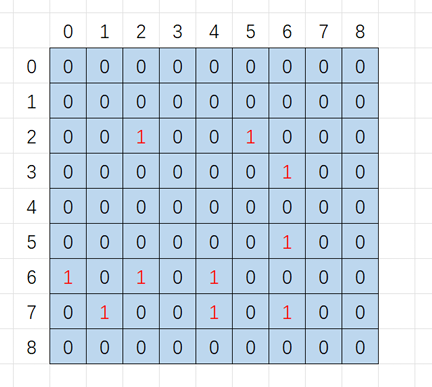
（4）默认随机布置10个雷

（5）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。

1.2 分析

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。

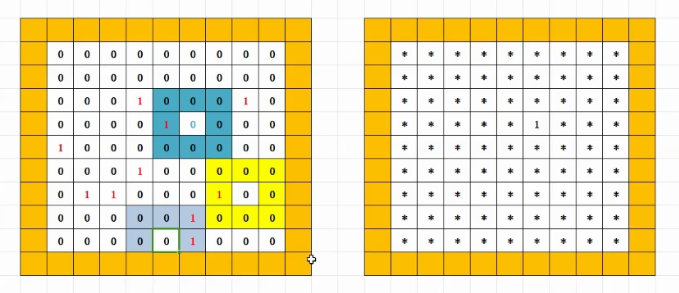
如果这个位置布置雷，我们就存放1，没有布置雷就存放0。



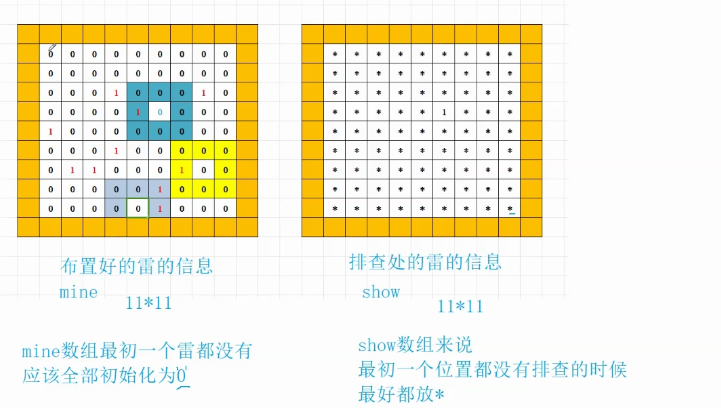
当我们排查雷的时候，如果排查到（3，6）坐标，如果这个坐标是雷就被炸死，如果不是雷就显示周围雷的个数，会存在歧义，是雷还是排查出来的雷？这里我们肯定有办法解决，⽐如：雷和非雷的信息不使用数字，使用某些字符，这样就避免冲突了，但是这样做棋盘上有雷和⾮雷的信息，还有排查出的雷的个数信息，就⽐较混杂，不够方便。

这里我们专门给一个棋盘（mine数组）存放布置好的雷的信息，再给另外一个棋盘（show数组）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。

假设当我们排查到（8，4）这个坐标时，我们访问周围8个数据，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 ⽐较合适。



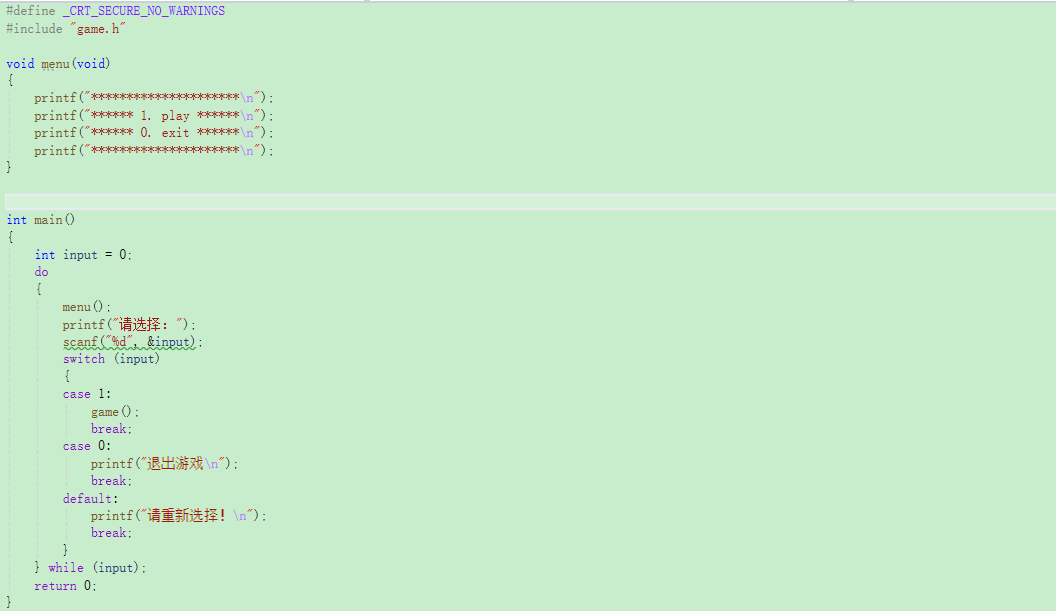
同时为了保持神秘，show数组开始时初始化为字符'\*'，为了保持两个数组的类型一致，可以使用同一套函数处理，mine数组最开始也初始化为字符'0'，布置雷改成字符'1'。



使用字符“1”和字符“0”的优势：计算雷的个数比较方便。

2、代码实现

test.c：



game.c：

