#include<stdio.h>

#include<stdlib.h>

#include<time.h>

//定义棋盘的大小

#define MAX\_ROW 10

#define MAX\_COL 10

#define DEFAULT\_LEIZI\_COUNT 10

int Menu(){

printf("=======================\n");

printf("1.开始游戏\n");

printf("0.退出游戏\n");

printf("=======================\n");

int choice = 0;

scanf("%d", &choice);

return choice;

}

void Init\_map(char show\_map[MAX\_ROW + 2][MAX\_COL + 2],

char mine\_map[MAX\_ROW + 2][MAX\_COL + 2]){

for (int row = 0; row < MAX\_ROW + 2; row++){

for (int col = 0; col < MAX\_COL + 2; col++){

show\_map[row][col] = '\*';

}

}

for (int row = 0; row < MAX\_ROW + 2; row++){

for (int col = 0; col < MAX\_COL + 2; col++){

mine\_map[row][col] = '0';

}

}

int mine\_count = DEFAULT\_LEIZI\_COUNT;

while (mine\_count>0){

int row = rand() % 10 + 1;

int col = rand() % 10 + 1;

if (mine\_map[row][col] == '1'){

//此处有雷了

continue;

}

mine\_map[row][col] = '1';

--mine\_count;

}

}

void Print\_map(char map[MAX\_ROW + 2][MAX\_COL + 2]){

printf(" ");

for (int i = 1; i <= MAX\_ROW; i++){

printf("%d ", i);

}

printf("\n");

for (int i = 1; i <= MAX\_ROW; i++){

printf("---");

}

printf("\n");

for (int row = 1; row <= MAX\_ROW; row++){

printf("%02d |", row);

for (int col = 1; col <= MAX\_COL; col++){

printf("%c ", map[row][col]);

}

printf("\n");

}

}

void Updateshowmap(char show\_map[MAX\_ROW + 2][MAX\_COL + 2],

char mine\_map[MAX\_ROW + 2][MAX\_COL + 2],int row,int col){

//此处是当我们点击一个下雷处后，我们要判断此处四周有多少个雷子

int mine\_lei = (mine\_map[row - 1][col - 1] - '0') +

(mine\_map[row][col - 1] - '0') +

(mine\_map[row + 1][col] - '0') +

(mine\_map[row - 1][col + 1] - '0') +

(mine\_map[row - 1][col] - '0') +

(mine\_map[row][col + 1] - '0') +

(mine\_map[row + 1][col + 1] - '0') +

(mine\_map[row + 1][col - 1] - '0');

show\_map[row][col] = mine\_lei + '0';

}

void Game(){

//首先我们要定义两个数组，一个用于展现来看，当前的情况，另一个用于玩家使用落子

char show\_map[MAX\_ROW + 2][MAX\_COL + 2];

char mine\_map[MAX\_ROW + 2][MAX\_COL + 2];

//我们定义的是10\*10的一个棋盘，但是我们在此处加上一个边框，上下左右各一个，所以我们需要+2

//接下来的步骤我们跟三子棋比较类似，初始化与打印棋盘都是我们一开始的工作。

//1.对数组初始化。初始化对两个棋盘都需要布置，之后把雷子随机的加入到地图中

Init\_map(show\_map,mine\_map);

//2.打印初始地图

Print\_map(show\_map);

int leizi\_count = 0;

while (1){

//3.让玩家输入坐标（并判定坐标是否合法）

//4.判定玩家是否踩雷，如果踩雷，gg

//5.如果没踩雷，判定是否掀开了所有格子（玩家胜利）

//6.就统计该位置周围有几个雷，并且把数字更新到地图上

int row, col;

scanf("%d %d", &row, &col);

if (row <= 0 || row > MAX\_ROW || col <= 0 || col > MAX\_COL){

printf("你的坐标是不准确的\n");

continue;

}

if (mine\_map[row][col] == '1'){

printf("此处有雷\n");

Print\_map(mine\_map);

break;

}

leizi\_count++;

if (leizi\_count == MAX\_ROW \* MAX\_COL - DEFAULT\_LEIZI\_COUNT){

printf("扫雷成功\n");

Print\_map(mine\_map);

break;

}

Updateshowmap(show\_map,mine\_map,row,col);

Print\_map(show\_map);

}

}

void start(){

while (1){

int choice = Menu();

if (choice == 0){

break;

}

Game();

}

}