**专题一：结构化程序设计与递归函数**

问题1：走迷宫问题

问题2：数独游戏问题

问题3：整数划分问题

组号：21 教师：张引、田沈晶、熊海辉

成员：

学号 3160103848 姓名 陈稼诚

学号 3170105860 姓名 彭子帆

**报告提交时间：2018.3.20**

# 问题1名称（完成的其他问题也按此结构模板撰写）

走迷宫问题

## 1.1 问题描述

## 用递归求解迷宫问题

## （迷宫示意图如右所示）

## 搜索一条从入口到达出口的可行路径

## 红色为围墙蓝色为可行通道

## 1.2 问题解决思路和关键点

### 1.2.1如何定义迷宫地图？

定义一个二维数组，用0代表墙，用1代表可走的路，用2代表走过的路径来记录路径

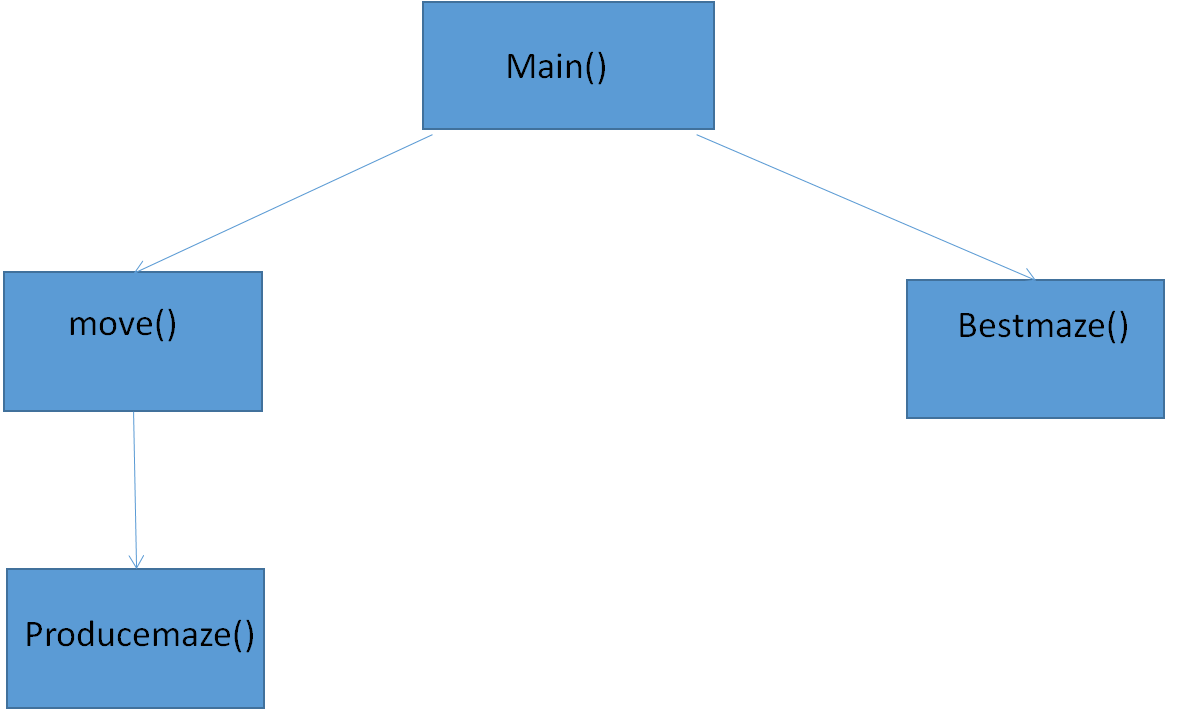
### 1.2.2如何在非图形模式下通过字符显示迷宫和路径？（若需要可继续分节）

用“@”代表墙，用“空格”代表可走的路，用“\*”来记录走过的路径。

## 1.3 程序结构

### 1.3.1 函数调用关系图

画出main函数及其下属各层调用的函数



### 1.3.2 全局变量说明

1）定义整型count

记录迷宫可行解的数量

2）定义整形min

记录最短路径长度min，初始值设定较大。

3）定义整型best\_count

记录迷宫最优解的个数

### 1.3.3 函数说明

1）函数原型：void move(int \*\*a,int \*\*best,int x,int y,int exit\_x,int exit\_y,int length,int width)

递归函数

功能描述：寻找迷宫的解

参数描述： 首个二维数组a[][]为迷宫的地图，best[][]为迷宫的最优解，x,y记录当前位置，exit\_x, exit\_y为迷宫出口位置，length, width为迷宫的长和宽。

返回值描述：本函数无返回值

重要局部变量定义：整型i

重要局部变量用途描述：用于循环

函数算法描述：首先标记当前位置，判断是否走到终点。若是则输出迷宫的解，若否则利用循环以右下左上顺序判断是否为可行通道。若可以的递归处理下一个单元。循环结束将当前位置在地图上标记改为路径标记。

2）函数原型： void produce\_maze(int \*\*a,int \*\*best,int length,int width)

功能描述： 打印迷宫

参数描述： 第一个二维数组a[][]记录迷宫地图，best[][]为迷宫的最优，length, width为迷宫的长和宽。

返回值描述：本函数无返回值

重要局部变量定义：整型step

重要局部变量用途描述：用于记录当前迷宫的步数便于判断最优解。

函数算法描述：利用0，1，2标记判断迷宫的解并输出同时记下解的长度，和已知最优解比较，若小于，则将此解地图存入best[][]。

3）函数原型：void best\_maze(int \*\*best,int length,int width)

功能描述： 打印最优解个数并输出一种最优解。

参数描述： best[][]为最优解地图，length, width为迷宫的长和宽。

返回值描述：本函数无返回值。

重要局部变量定义：循环变量i，j；

函数算法描述：输出全局变量best\_count的值，并利用循环打印best[][]中的迷宫地图。

4）函数原型：int main()

功能描述： 连接各个函数。负责输入，定义变量、动态数组，讲数组传进解决迷宫函数并调用最优解判断函数。

参数描述：无

返回值描述：无意义

重要局部变量定义：整型i,j用于循环，整型 enter\_x,enter\_y,exit\_x,exit\_y为出入口坐标整型length,width为输入迷宫的长与宽。整型flag用于标记输入迷宫是否合法。

函数算法描述：先利用malloc函数分配动态二维数组存储迷宫，在判断输入合法后将迷宫传进走迷宫函数。最后调用迷宫最优解函数，并释放分配的空间。

### 1.3.4 程序文件结构

1）文件函数结构

文件分为main.c, 输出迷宫函数.c, 走迷宫函数.c, fun.h.

输出迷宫函数.c中包含produce\_maze()函数以及best\_maze函数。

走迷宫函数.c中包含move（）函数。

Fun.h中包含三者函数原型。

2）多文件构成机制

在输出迷宫函数中，定义了外部变量count，min，best\_count（在main函数中第一次定义）。

在fun,h中用ifndef防止重复定义

说明分文件构成程序的实现机制（即如何具体采用文件包含、#define保护、外部变量或外部函数）

## 1.4 安装运行说明

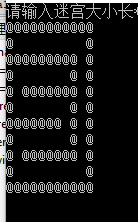
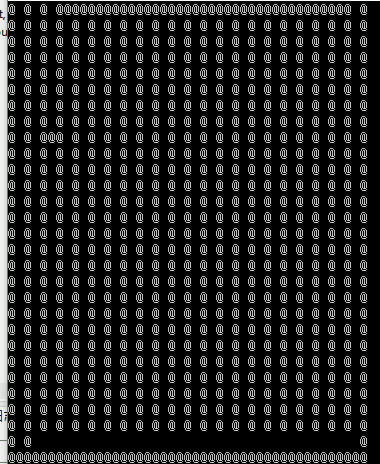
运行时现将main.c、输出迷宫函数.c走迷宫函数.c文件编译为.o文件，再将.o文件以及头文件串联起来生成可执行的exe文件.

## 1.5 总结

### 1.5.1 实践过程中遇到的难点及解决方案

二位动态数组的分配问题以及使用数组时出现了程序崩溃的情况。后经讨论与试验后发现递归函数中存在遭遇边界后继续递归的问题。修改后即可使用二维数组。

在写完函数迷宫的主体以后，彭子帆着手思考了能否附加生成迷宫函数。在撰写生成迷宫函数的过程中，使用的随机数函数不够随机导致出现了如下生成迷宫情况：



### 1.5.2 程序亮点或基本功能之外的拓展点

能动态地输入迷宫大小，入口出口以及。

能输出多种走法，并且记录最优解。

### 1.5.3 实践之不足

不会图形界面，不能自己随机生成迷宫。

### 1.5.4 分工合作和各自的收获

一起探讨了不同种算法的走迷宫方式，思维的碰撞带来了视野的拓展，也学会了如何写接口文件，多文件程序。最重要的是理解了如何合作完成一个大型的程序。

# 问题2名称

数独游戏问题

## 问题描述

## 问题：数独游戏是一个常见的游戏，玩法是一个9\*9的矩阵，其中的每个格子都放有一个数字，保证每一行、每一列、每个3\*3的范围内都含有1-9的数，并且这些数不能重复。

## 输入：一个9\*9的矩阵，里面随机填了一些数，没有填的空格用0表示。

## 输出：被解出来的数独

## 2.2 问题解决思路和关键点

### 2.2.1 记录一行、一列、小九宫格的数（若需要可继续分节，下同）

通过二维数组很轻松记录一行一列数字，然后进行判断。

再找到每个小九宫格的首个数的坐标与第n个小九宫格的关系，进行递归可以记录小九宫格的数字。

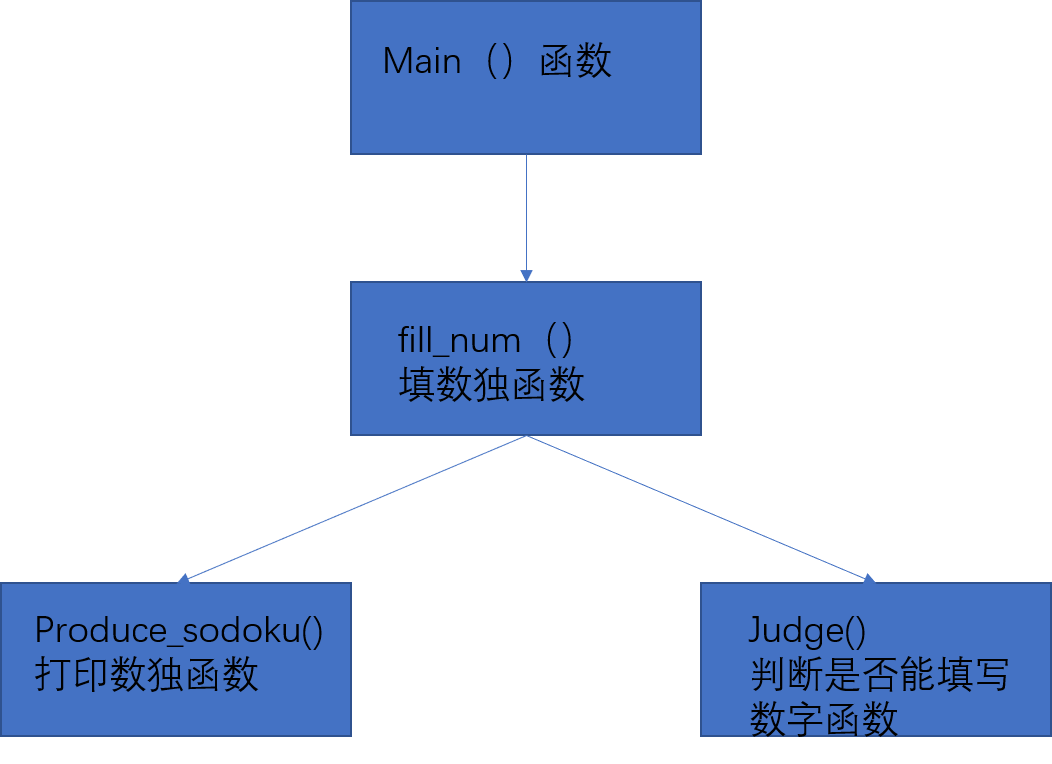
### 2.2.2 判断是否成立并进行递归填下一个数字（若需要可继续分节）

记录后就很容易通过循环判断出该数独是否成立，若成立则递归填下一个数字。

## 2.3 程序结构

### 2.3.1 函数调用关系图

画出main函数及其下属各层调用的函数



### 2.3.2 全局变量说明

1）二维数组a[9][9]

存储输入数独的数组

2）整型count

记录数独的解得变量

### 2.3.3 函数说明

1）函数原型：void fill\_num(int x, int y)(递归函数)

功能描述；递归实现填写数独的功能

参数描述： x, y为输入坐标。函数从此点开始执行。

返回值描述：此函数无返回值

重要局部变量定义：整型i

重要局部变量用途描述：用于内部循环

函数算法描述：

2）函数原型：void produce\_sudoku(void)

功能描述；打印解完的数独

参数描述： 无

返回值描述：无

重要局部变量定义：整型i , j,

重要局部变量用途描述：用于循环输出数组

函数算法描述：利用双层循环输出全局变量二维数组的值。

int judge(void)

3）函数原型：int judge(void)

功能描述；判断是否可以在此格中填写数字。

参数描述： 无

返回值描述：若可以填写此数字填写则返回1，否则分返回0

重要局部变量定义：整型i , j, k, n, 数组a[].

重要局部变量用途描述：ijk用于循环，n用于赋值，数组a[]用于存储小九宫格中的数字。

函数算法描述：简单的利用三个循环判断行列以及小九宫格中是否存在该数字。

4）函数原型：int main(void)

功能描述；串联所有函数

参数描述： 无

返回值描述：无具体意义

重要局部变量定义：整型i , j, flag.

重要局部变量用途描述：I, j用于循环，flag用于标记输入是否合法

函数算法描述：利用循环判断输入是否合法，然后调用解决数独函数。

### 2.3.4 程序文件结构

1）文件函数结构

本程序包含main.c, 填数独函数.c, 判断数独函数.c, 判断函数.c, bool.h, 五个文件。

其中main.c中为main函数。填数独函数.c包含了fill\_num填写数独函数。输出数独函数.c包含了produce\_sodoku函数，判断函数.c包含了judge函数。Bool.h定义了上面三个函数以及true与false的值。

2）多文件构成机制

说明分文件构成程序的实现机制（即如何具体采用文件包含、#define保护、外部变量或外部函数）

## 2.4 安装运行说明

运行时现将main.c, 填数独函数.c, 判断数独函数.c, 判断函数.c编译为.o文件，再将.o文件以及.h文件串联起来生成可执行的exe文件.

## 2.5 总结

### 2.5.1 实践过程中遇到的难点及解决方案

因为有较多情况，所以程序无法采用走迷宫函数简单的回溯法，只能采取更普遍的循环式的回溯法。

### 2.5.3 实践之不足

此程序无法自己生成数独的拓展功能。在写程序的时候没有想到解决此问题更好的办法。此程序运行时间较长，并不是一个非常高效的算法。

### 2.5.4 分工合作和各自的收获

双方有交流程序的算法。程序主要由彭子帆完成，并使用了他的算法。我们在撰写代码的过程学习总结了递归的算法，并提升了撰写较长的代码的能力。

# 3. 感想

三个问题做下来还是稍微有些难度的，写完两个递归程序感觉对递归的了解更深入了一步。和彭子帆的合作也让我受益匪浅。让我这个非本专业的学生感受到了真正的编程。