**数据结构上机报告**

**学 校： 中国地质大学（武汉）**

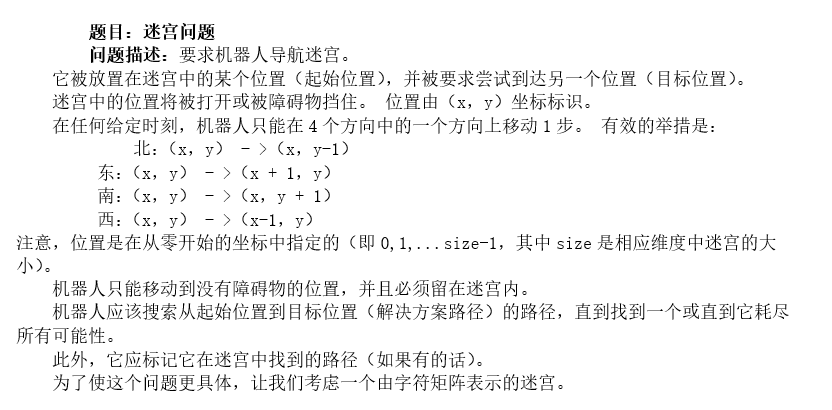
**专 业： 电子信息**

**班 级： 071183**

**姓 名： 程子轩、李玮樨**

**完成时间： 2020/11/15**

1. 题目描述



根据题目描述可知，任务目标是找出迷宫起点到终点的坐标路径，根据要求，我们需要如下几个模块：

1. 模块划分
2. 文件读入模块

文件读入模块用于读入地图的配置文档信息并对地图的起点坐标、终点坐标、通路坐标进行读取并储存。

1. 地图生成模块

地图生成模块根据文件读入模块读到的配置文件信息进行地图（二维数组）的生成，并在生成的二维数组中进行起点、终点、通路、非通路的标示。

1. 寻路模块

寻路模块即从起点开始，进行寻路操作，当寻找到终点或无法走到终点时停止。

1. 结果输出

当寻路模块到达终点时，便可以通过此模块从终点开始进行逆向的路径追寻并输出整条路径。

为实现以上模块，我们需要定义地图节点的数据结构：

1. 数据结构

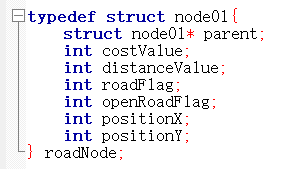
为确保能从终点逆向得到整条路径，地图节点中需要储存其上一个节点的地址信息（即是从哪一个节点来到这的）。

为确保路线的行进大体上是朝向终点，地图节点中需要储存一个根据其与终点距离的关系而得到的权值。

为分辨该节点是否是可行走的通路，地图节点需要一个标志位。

由于地图二维数组用于存储地图节点的指针，即我们无法很好地根据地图节点得到其在地图中的位置，因此地图节点中需要存储其在地图中的位置信息。

综上所述，地图节点的结构长这个模样：



其中：

parent 为上一节点的指针；

costValue为行进至该格所需的花费，在只能在邻近四格内寻路的情况下并无实际作用，仅用于将来可能的邻近八格寻路的程序扩展；

distanceValue为当前节点与终点间的距离权值，其计算公式为：



roadFlag用于标志该节点是否是通路

openRoadFlag用于标志该节点是否是可以行走的通路（例如一般情况下走过的路该位置零，经检验无法行进至终点的路该位均置零）。

positionX与positionY为该节点在地图中的坐标值。

根据地图节点的结构，地图的结构为存储着地图节点指针的二维数组：



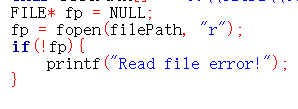
接下来根据模块来进行源代码的讲解：

1. 源程序
2. 文件读入部分

首先将文件地址以字符串的形式存储



要注意反斜杠\别被程序识别为转义符



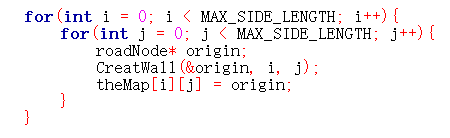
接下来用文件指针打开文件。

1. 地图生成部分

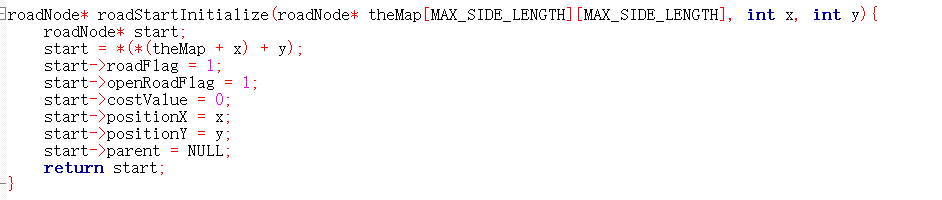
在读入节点信息前，我们要先将地图生成好。首先定义二维数组：

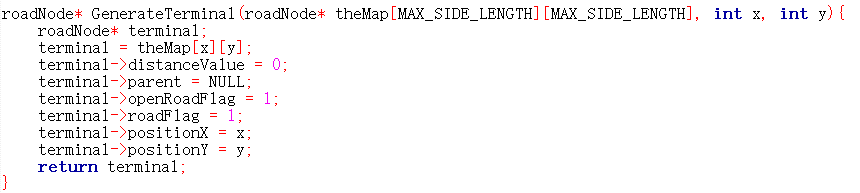


然后将地图中所有的节点初始化为不可通行的节点：

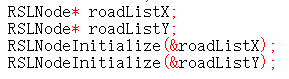


之后继续进行配置文件的读入，读入进来的终点信息和起点信息我们可以直接利用这两个函数进行初始化：



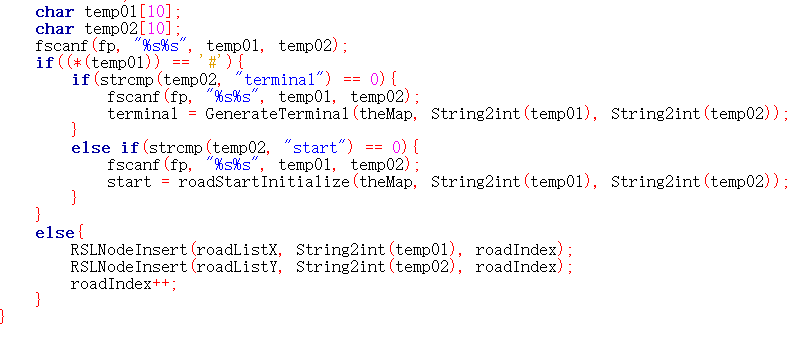


但读到的通路信息由于数量过大，不宜在读取中进行初始化，因此要定义两个链表用以存储通路信息：



最后在定义一个通路索引，用于指导链表元素的添加：

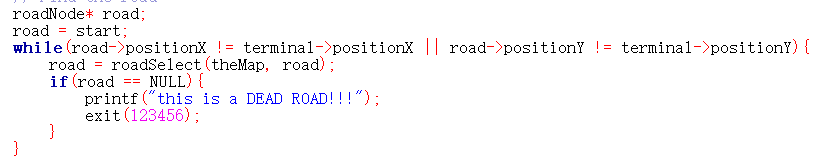




进行读入，并根据读入的配置对地图进行配置。

1. 寻路部分

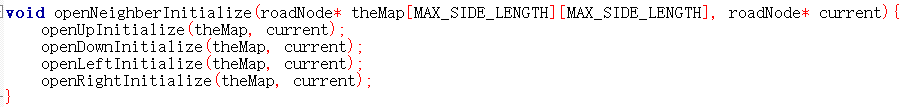
寻路时，我们只要定义一个活动地图节点指针并让它一开始指向地图起点，随后调用寻路函数并将返回值继续调用寻路函数直到寻路结束即可：

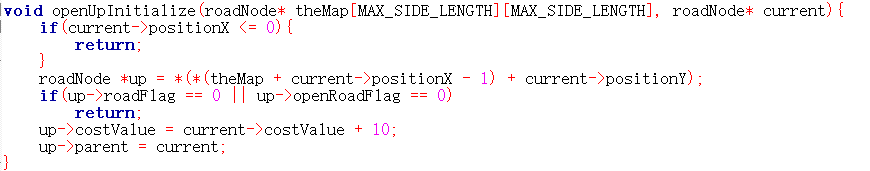


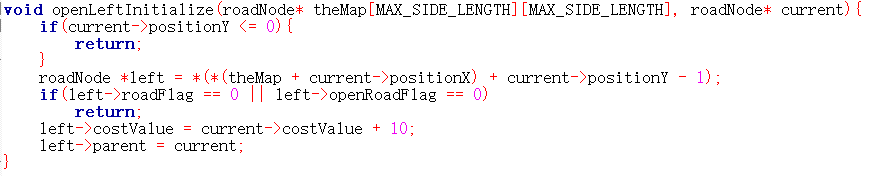
接下来让我们进入寻路函数

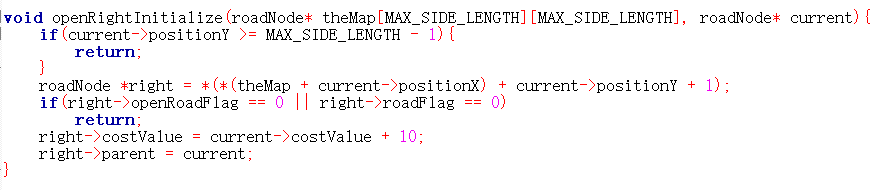


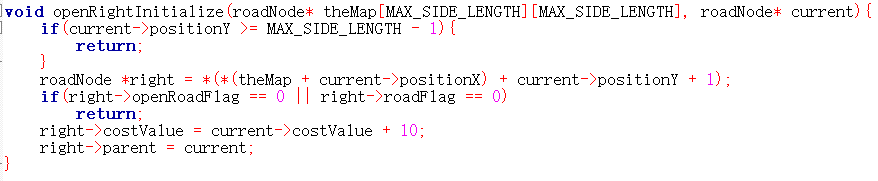
寻路函数的开端，我们需要将当前地图节点设为“已走过”并对该节点的四邻节点进行初始化（判断是否是可以通行的节点，如果是的话赋予通行费权值costValue与距离权值distanceValue，并将当前节点作为该邻节点的前置节点）：



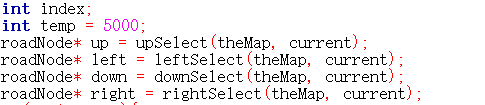


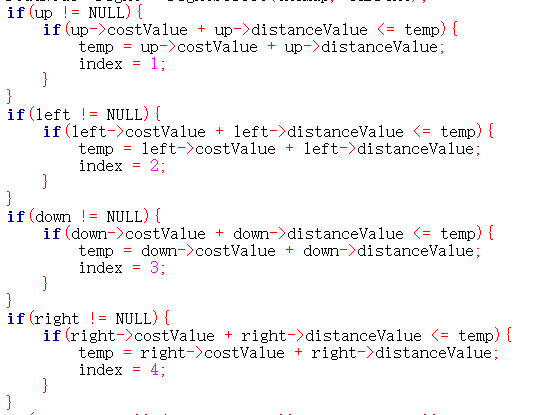




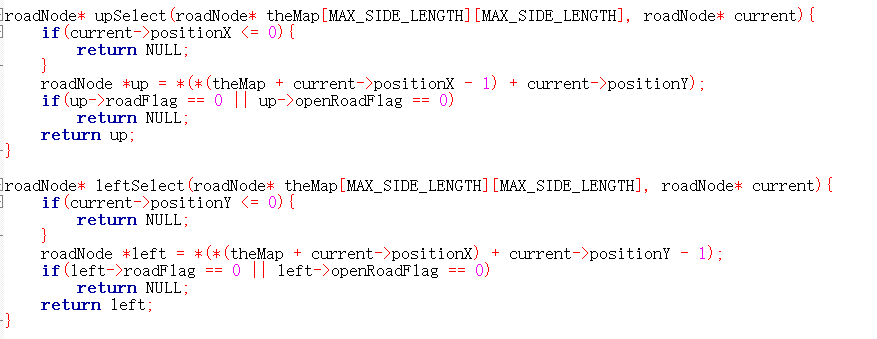


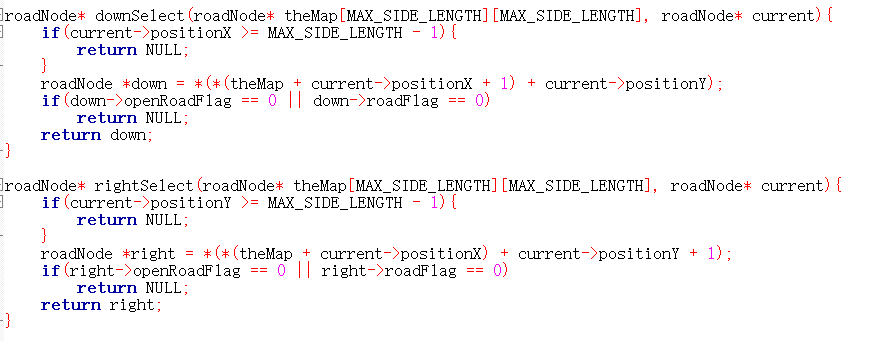
邻近节点初始化后，便进入道路选择环节：



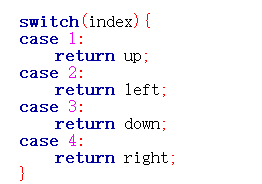


首先定义一个临时索引index与一个大值temp。index用于存储最终决定行走方向的标志，temp用于存储当前最小的行走总消耗权值。





方向选择函数分为边界判断与通路判断两部分，两部分判断均通过后，便认为该方向存在可以通行的节点，并返回该邻节点进行之后的最小花费权值筛选，最后的筛选结果即为当前节点道路选择的结果值：



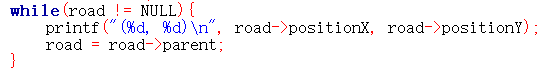
如果遇到走入死路的情况（因为每走一步都会将走过的地方标志为不可通行），则回到上一步进行重新判断：



至此，我们完成了寻路的全过程。

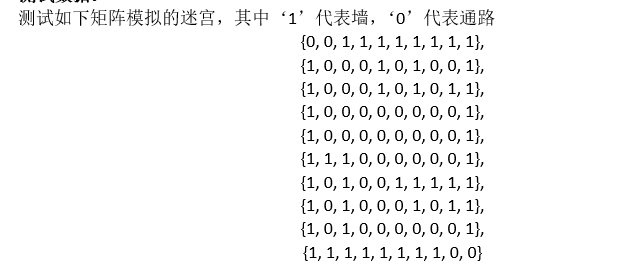
1. 结果输出部分

这个部分即从终点依次输出前置节点直至到达起点：

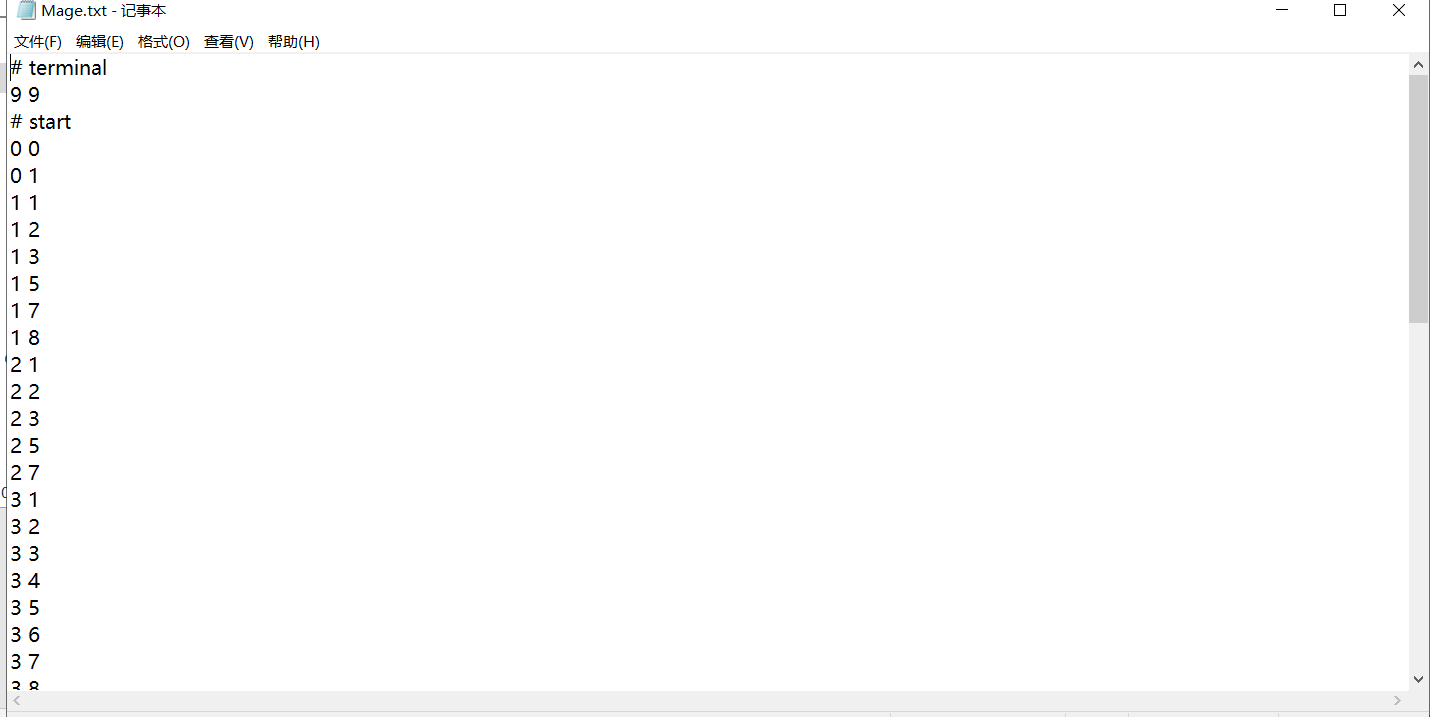


1. 结果展示

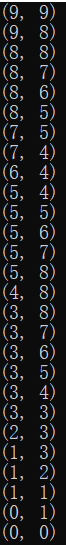
根据所要求的测试地图：



进行合乎格式的配置文件编辑：

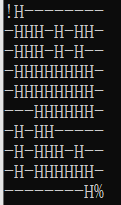


在此地图下得出的结果为：



从下到上便为起点到终点的路径。

程序输出的地图如下：



其中！为起点

H为通路

-为不通路

%为终点

与上面输出的路径信息进行对照可以发现正确无误。

本次实验圆满成功。