Solution of Maze

Programmer: 周晨恺

Test & Reporter: 郑楷

Date: 2015-03-21

Chapter 1: Introduction

接受一个二维迷宫和起止位置

用递归算法求解迷宫

由于两人都写了各自的版本，所以最终讨论后修改，合并为一个文件

核心部分采用 Programmer 的最短路径求法

（Report 写的是将所有状态直接记录在 map 上的只求第一匹配项的求法）

初始化方式和输出方式综合两人的程序，修改为由用户决定即接受用户指令——

初始化方式包括文件输入和手动输入两种

文件输入读取的是同目录下的 maze.txt 文件

格式为迷宫（0 为墙，1 为路）+起点+终点

输出方式包括路径输出和地图绘制两种

地图绘制中+表示墙，_表示路，@表示路径，$表示终点

Chapter 2: Algorithm Specification

用一个二维数组 map 来记录迷宫

用数组 step_x 和 step_y 来记录当前路径（其实也可以采用二维数组）

用数组 ans_x 和 ans_y 来记录目前已知的最短路径（也可以用二维数组）

并规定如下代号——

0：墙体

1：可通过的路

2：路径

3：终点

路径寻求采用回溯算法

并且遍历所有路径，比较步数找出最短路径

将运动方向的坐标增量记录到数组 nx 和 ny 中

通过循环来分别向四个方向移动，牺牲空间来简化代码

- 每次移动首先要判断是否是终点
  - 如果是终点，要比较已知最短路径的步数（初始化时已赋予整型的最大值）和当前路径的步数
    - 如果当前路径更短，则复制当前路径到最短路径中
  - 如果不是终点，就将这一步写入当前路径，并使用 0 占位，防止“进退进”和“田”

字型的死循环，然后进入四个方向可行性的判断

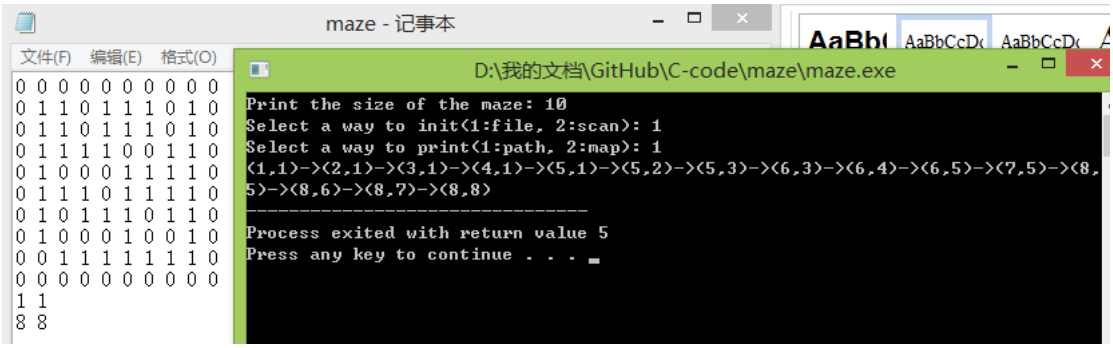
- 四个方向都尝试过后进行回溯，即取消这一步的占位状态（恢复为 0），并且在当前路径中将这一步归零

Chapter 3: Testing Results

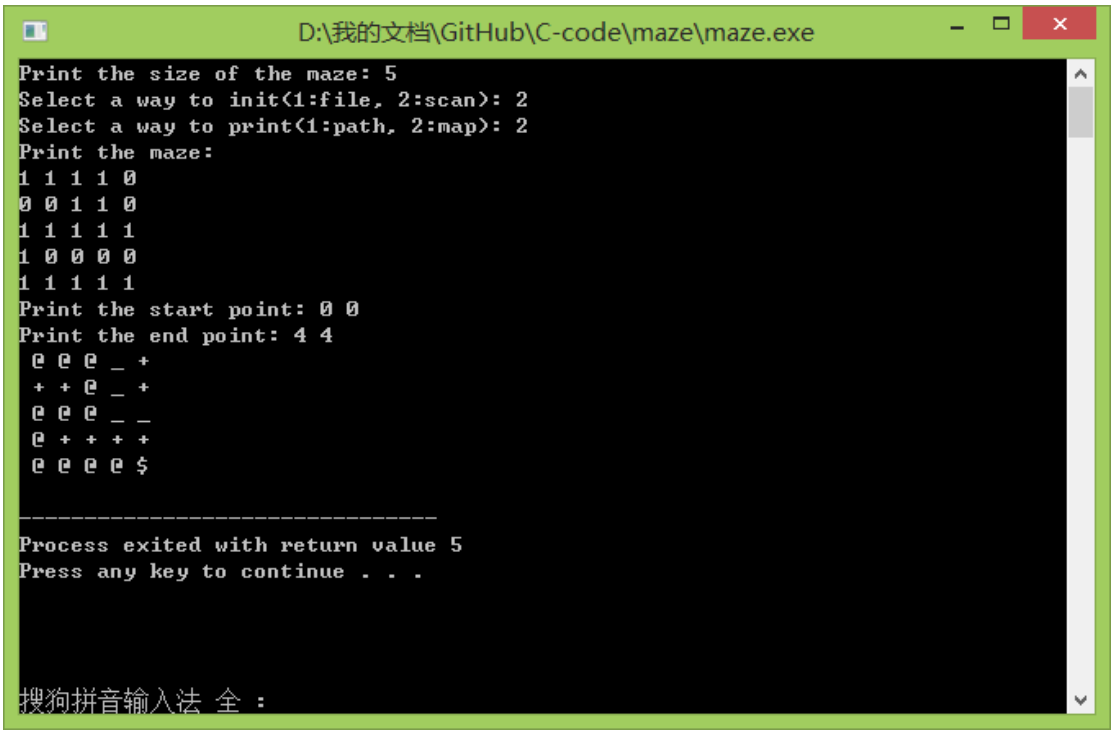
测试：主要使用 PPT 中的迷宫进行测试

功能测试：

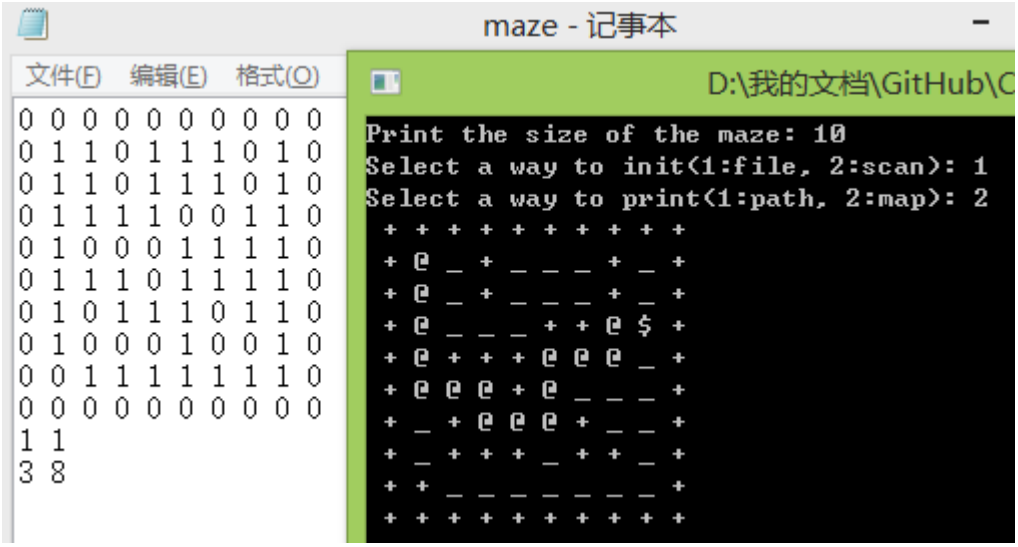
文件读取和路径输出：



手动输入和地图绘制：

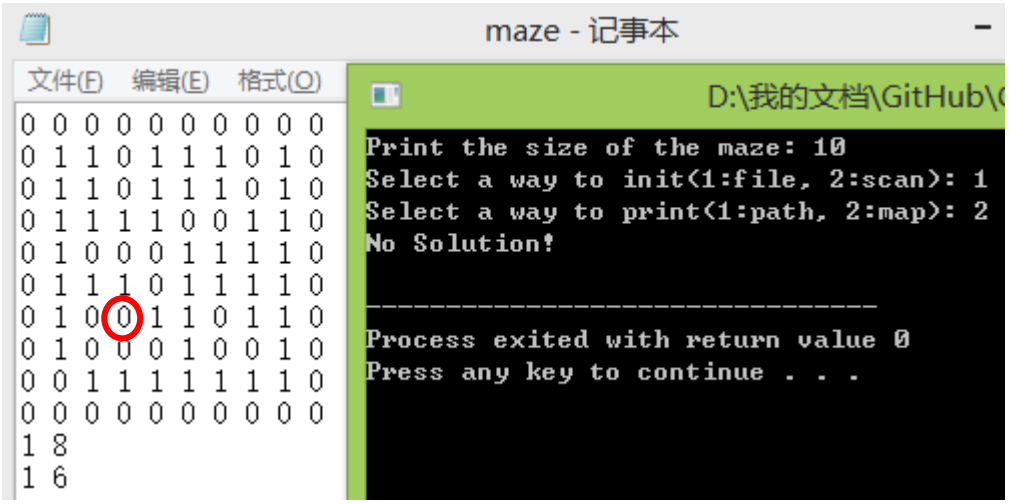


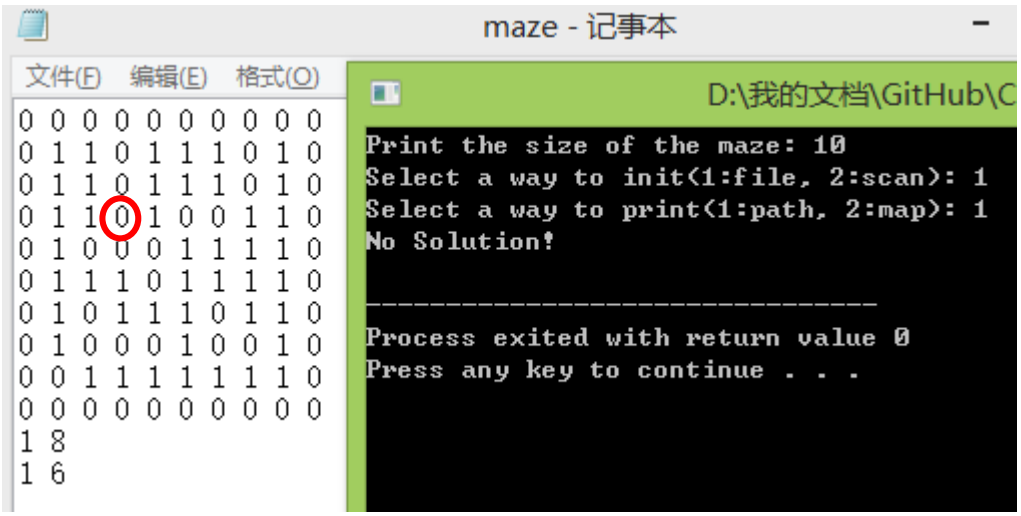
路径规划测试：（为了简化测试工作，都采用文件输入、地图绘制）


maze - 记事本
文件(F) 编辑(E) 格式(O)
0 0 0 0 0 0 0 0 0 0
0 1 1 0 1 1 1 0 1 0
0 1 1 0 1 1 1 0 1 0
0 1 1 1 1 0 0 1 1 0
0 1 0 0 0 1 1 1 1 0
0 1 1 1 0 1 1 1 1 0
0 1 0 1 1 1 0 1 1 0
0 1 0 0 0 1 0 0 1 0
0 0 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0
1 1
3 8
D:\我的文档\GitHub\C
Print the size of the maze: 10
Select a way to init(1:file, 2:scan): 1
Select a way to print(1:path, 2:map): 2
+ + + + + + + + + +
+ @ _ + _ _ _ + _ +
+ @ _ + _ _ _ + _ +
+ @ _ _ _ + + @ $ +
+ @ + + + @ @ @ _ +
+ @ @ @ + @ _ _ _ +
+ _ + @ @ @ + _ _ +
+ _ + + + _ + + _ +
+ + _ _ _ _ _ _ _ +
+ + + + + + + + + +


maze - 记事本
文件(F) 编辑(E) 格式(O)
0 0 0 0 0 0 0 0 0 0
0 1 1 0 1 1 1 0 1 0
0 1 1 0 1 1 1 0 1 0
0 1 1 1 1 0 0 1 1 0
0 1 0 0 0 1 1 1 1 0
0 1 1 1 0 1 1 1 1 0
0 1 0 1 1 1 0 1 1 0
0 1 0 0 0 1 0 0 1 0
0 0 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0
1 1
8 8
D:\我的文档\GitHub\C
Print the size of the maze: 10
Select a way to init(1:file, 2:scan): 1
Select a way to print(1:path, 2:map): 2
+ + + + + + + + + +
+ @ _ + _ _ _ + _ +
+ @ _ + _ _ _ + _ +
+ @ _ _ _ + + _ _ +
+ @ + + + _ _ _ _ +
+ @ @ @ + _ _ _ _ +
+ _ + @ @ @ + _ _ +
+ _ + + + @ + + _ +
+ + _ _ _ @ @ @ $ +
+ + + + + + + + + +


maze - 记事本
文件(F) 编辑(E) 格式(O)
0 0 0 0 0 0 0 0 0 0
0 1 1 0 1 1 1 0 1 0
0 1 1 0 1 1 1 0 1 0
0 1 1 1 1 0 0 1 1 0
0 1 0 0 0 1 1 1 1 0
0 1 1 1 0 1 1 1 1 0
0 1 0 1 1 1 0 1 1 0
0 1 0 0 0 1 0 0 1 0
0 0 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0
1 8
1 6
D:\我的文档\GitHub\C
Print the size of the maze: 10
Select a way to init(1:file, 2:scan): 1
Select a way to print(1:path, 2:map): 2
+ + + + + + + + + +
+ _ _ + @ @ $ + @ +
+ _ _ + @ _ _ + @ +
+ @ @ @ @ + + _ @ +
+ @ + + + _ _ _ @ +
+ @ @ @ + @ @ @ @ +
+ _ + @ @ @ + _ _ +
+ _ + + + _ + + _ +
+ + _ _ _ _ _ _ _ +
+ + + + + + + + + +





随机选取了(1,1)->(8,8)，(1,8)->(1,6)，(1,1)->(3,8)为测试点，结果都能找到最短路径

又人为的堵上一条路，path 和 map 的结果也如预期显示“No solution!”，结果正确

就以上测试结果而言，程序能够正常运行，达到预期目标

Chapter 4: Analysis and Comments

程序能够正确运行，找到迷宫正确的最短路径，但是仍有不足——

1、程序采用递归，效率不高，尤其在迷宫规模足够大的时候并不能找出正确路径

2、程序全局变量过多

3、为了实现自定义规模的迷宫，程序一开始就定义了 100*100 的迷宫以及 4 个记录路径的 100 的数组，很大程度浪费了空间；事实上可以通过堆栈，动态分配空间，来满足规模自定义的要求，节省更多的空间