二、课程设计内容(含技术指标)

【问题描述】

以一个 mXn 的长方阵表示迷宫，0 和 1 分别表示迷宫中的通路和障碍.设计一个程序，对任意设定的迷宫，求出一条从入口到出口的通路,或得出没有通路的结论。

【任务要求】

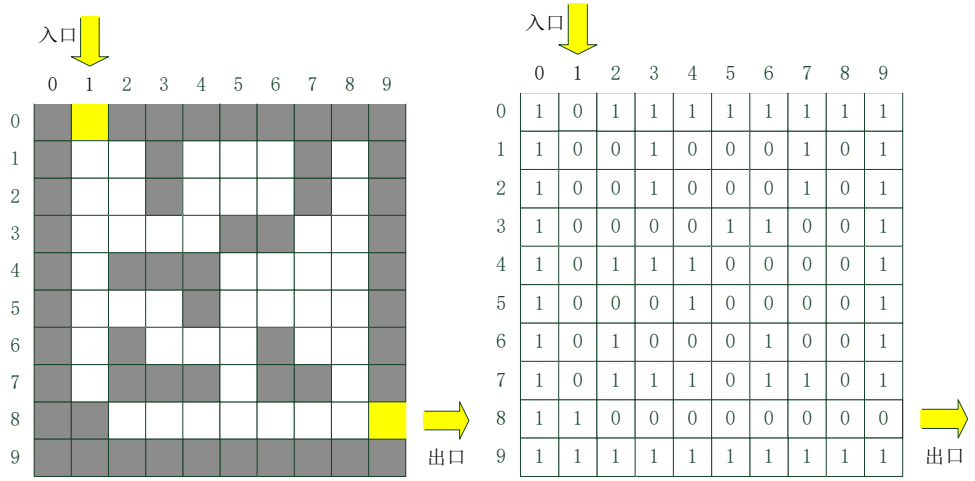
首先实现一个以链表作存储结构的栈类型,然后编写一个求解迷宫的非递归程序。求得的通路以三元组（i，j，d）的形式输出。其中:（i，j）指示迷宫中的一个坐标，d 表示走到下一坐标的方向.如,对于下列数据的迷宫，输出一条通路为:（1，1，1)，（1，2，2)，（2，2，2），（3，2，3)，(3,1，2)，…。

编写递归形式的算法,求得迷宫中所有可能的通路。

以方阵形式输出迷宫及其通路。

【测试数据】

迷宫的测试数据如下：左上角(0，1）为入口，右下角（8,9）为出口。



	0	1	2	3	4	5	6	7	8	9
0	1	0	1	1	1	1	1	1	1	1
1	1	0	0	1	0	0	0	1	0	1
2	1	0	0	1	0	0	0	1	0	1
3	1	0	0	0	0	1	1	0	0	1
4	1	0	1	1	1	0	0	0	0	1
5	1	0	0	0	1	0	0	0	0	1
6	1	0	1	0	0	0	1	0	0	1
7	1	0	1	1	1	0	1	1	0	1
8	1	1	0	0	0	0	0	0	0	0
9	1	1	1	1	1	1	1	1	1	1

四、基本要求

1.在设计时,要严格按照题意要求独立进行设计,不能随意更改.若确因条件所限,必须要改变课题要求时，应在征得指导教师同意的前提下进行.

2．在设计完成后，应当场运行和答辩，由指导教师验收，只有在验收合格后才能算设计部分的结束.

3．设计结束后要写出课程设计报告，以作为整个课程设计评分的书面依据和存档材料。设计报告以规定格式的电子文档书写、打印并装订，报告格式严格按照模板要求撰写,排版及图、表要清楚、工整。

从总体来说,所设计的程序应该全部符合要求，问题模型、求解算法以及存储结构清晰;具有友好、清晰的界面;设计要包括所需要的辅助程序，如必要的数据输入、输出、显示和错误检测功能;操作使用要简便; 程序的整体结构及局部结构要合理;设计报告要符合规范。

课程负责人签名：

2011 年 7 月 1 日

迷宫与栈问题

摘　　要

数据结构是研究与数据之间的关系,是互相之间一种或多种特定关系的数据元素的集合,我们称这一关系为数据的逻辑结构。数据结构在计算机的表示（又称映像）称为数据的物理结构，又称存储结构.

本次课程设计是迷宫求解问题，主要是模拟从入口到出口的通路。程序中的数据采取的是“栈”作为数据的逻辑结构，并且使用链式存储结构，即是实现一个以链表作存储结构的栈类型.本课程设计实现了链栈的建立，入栈，出栈，判断栈是否为空的方法,关键的是迷宫通路路径的“穷举求解”和递归求解的方法.

本课程设计重要说明了系统的设计思路、概要设计以及各个功能模块的详细设计和实现方法。

本次程序的开发工具是 microsoftvisualstudio2008，编程语言是 C 语言。

关键词:迷宫求解　链栈　穷举求解　递归求解

目　录

1　需求分析

1。1 基本原理分析

迷宫问题通常是用“穷举求解”方法解决,即从入口出发，顺着某一个方向进行探索，若能走通，则继续往前走; 否则沿着原路退回，换一个方向继续探索，直至出口位置，求得一条通路。假如所有可能的通路都探索到而未能到达出口，则所设定的迷宫没有通路。栈是一个后进先出的结构，可以用来保存从入口到当前位置的路径。

定义迷宫类型来存储迷宫数据，通常设定入口点的下标为（1,1），出口点的下标为(n,n).为处理方便起见，在迷宫的四周加一圈障碍。对于迷宫任何一个位置，均约定东、南、西、北四个方向可通。

1。2 功能要求

(1)以一个 mXn 的长方阵表示迷宫，0 和 1 分别表示迷宫中的通路和障碍。迷宫的四周有一圈障碍。

（2）程序输出的结果以三元组（i，j,di）的形式输出,其中：（i,j）指示迷宫中的一个坐标，di 表示走到下一坐标的方向，di 的取值为 1、2、3、4 分别表示东、南、西、北

（3）程序能够输出一个任意的迷宫从指定入口到出口的所有通路，以及以方阵形式输出迷宫

（4）若设定的迷宫存在通路，则以方阵形式将迷宫及其通路输出到标准输出文件上，其中字符“1”表示障碍,“2”表示路径,“3”表示曾途经该位置但不能到达出口，其余位置用 0 表示。若设定迷宫不存在通路则报告相应信息

2　概要设计

2。1 数据结构及其抽象数据类型的定义

2.1.1 栈的抽象数据类型

ADT LinkStack {
数据对象：D={ai| ai∈CharSet,i=1，2…n，n〉=0}
数据关系：R1={〈ai-1， ai 〉 | ai—1, ai∈D，i=2，…n}
基本操作：
InitLinkStack(&S)
操作结果:构造一个空栈 S。
LinkStackEmpty(&S)
初始条件：栈 S 已存在。
操作结果:若 S 为空栈，则返回 TRUE，否则返回 FALSE。
PushLinkStack （&S，e）
初始条件：栈 S 已存在。
操作结果：在栈 S 的栈顶插入新的栈顶元素 e。
PopLinkStack （&S，&e）
初始条件：栈 S 已存在。
操作结果：删除 S 的栈顶元素，并以 e 返回其值.
} ADT LinkStack

2.1.2 迷宫的抽象数据类型

ADT Maze {
数据对象：D={ai,j| ai，j∈{0,1,2,3}，0〈=i<=m—1，0〈=j〈=n-1，m,n〈=10}
数据关系：R= {row，line}
基本操作：
InitMaze（&maze）
初始条件:用迷宫类型 Maze.arr[row][line]表示迷宫,迷宫的数据由用户自己定义,并且以值 0 表示通路，以值 1 表示障碍。
操作结果：构成迷宫的数字型数组，以 0 表示通路，以 1 表示障碍。
MazePath（&maze，start,end）
初始条件：迷宫 S 已被赋值。
操作结果：若迷宫 S 中存在一条通路，则按如下规定改变迷宫 S 的状态：以 2 表示路径上的位置,以 3 表示死胡同; 否则迷宫的状态不变.
MazePath_Recursion(&maze,cur,end,curstep)
初始条件：迷宫 S 已被赋值。
操作结果：若迷宫 S 中存在通路,输出迷宫的可能路径。
PrintMaze（Maze)
初始条件:迷宫 M 已经存在
操作结果：以方阵形式输出迷宫.

```
}ADT Maze;
```

2.1。3 功能模块分解

```
（1） 主程序模块：
void main()
{
   初始化；
   While(1) {
       接受命令；
       处理命令；
   }
}
```

（2) 栈模块————————实现栈的抽象数据类型

```
typedef struct Stacknode{
   SElemType data;
   struct Stacknode *next;
} *LinkStack;
```

调用函数:InitLinkStack （), LinkStackEmpty （）,PushLinkStack(), PopLinkStack()

（3）迷宫模块----实现迷宫抽象数据类型

```
typedef struct{
    int row;
   int line;
} PosType; //迷宫中row行line列的位置

typedef struct MazeType {
   int row;
    int line;
    int arr[MAXLEN] [MAXLEN] ;
} MazeType;    //迷宫类型

typedef struct{
   int ord; // 当前位置在路径上的“序号”
   PosType seat;  //当前的坐标位置
   int di; //往下一坐标位置的方向
}SElemType;
```

调用函数:
Pass(), FootPrint(), MarkPrint(), NextPos （）,MazePath()或 MazePath_Recursion (),PrintMaze ()。
各个模块之间的调用关系如下：
主程序模块—>迷宫模块 -〉栈模块

3　详细设计

3.1 主函数与各功能模块

主函数的各函数调用关系如图 3—1 所示,主函数调用创建迷宫函数和求解所有路径的函数,其中创建迷宫信息的函数调用初始化迷宫和输出迷宫.

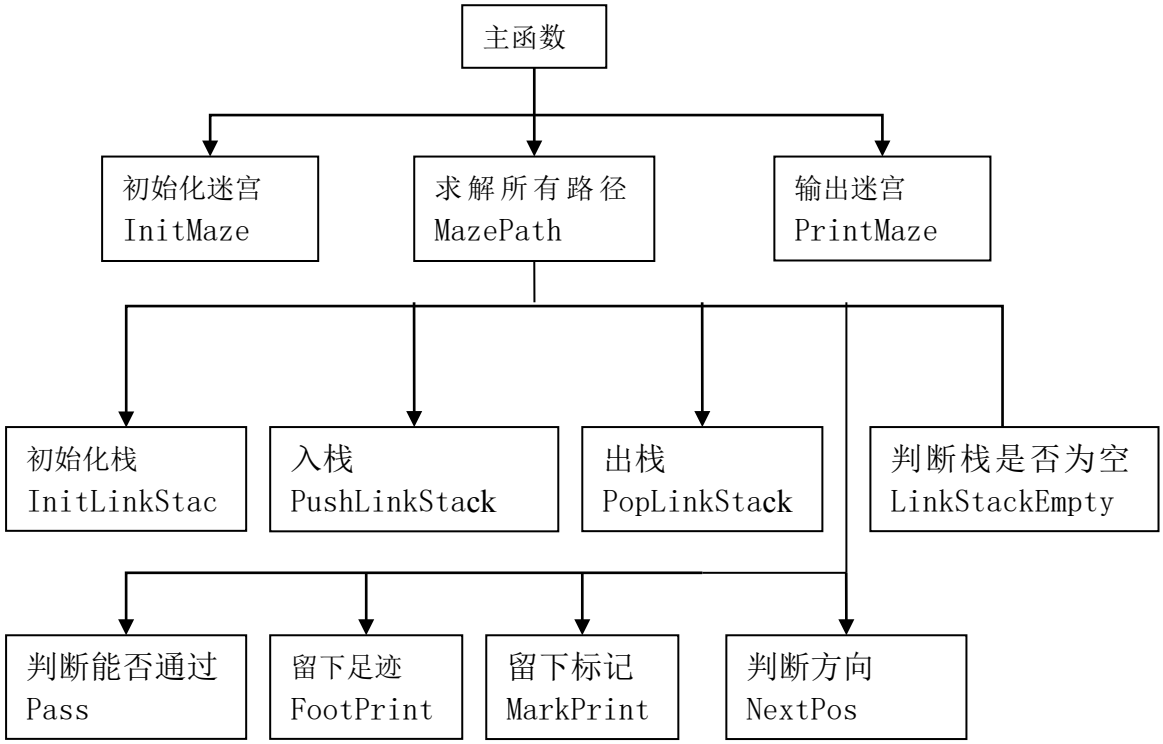


图 3-1

3。2 迷宫路径模块

3.2。1 算法分析

解决迷宫问题最重要的程序段是找到通路,并存储在栈里，现只分析实现这一过程的函数

```
do {
   若当前位置可通，
   则{    将当前位置插入栈顶;
          若该位置是出口位置，则结束;
          否则切换当前位置的东邻方块为新的当前位置;
     }
   否则,
```

若栈不空且栈顶位置尚有其他方向未经探索，
则设定的新的当前位置为沿顺时针方向旋转找到的栈顶位置的下一相邻块；
若栈不空但栈顶位置的四周均不可通，
则 {　　删去栈顶位置；
若栈不空，则重新测试新的栈顶位置，
直到找到一个可通的相邻块或出栈至栈空；
}
}while（栈不空）；

3。2。2 流程图

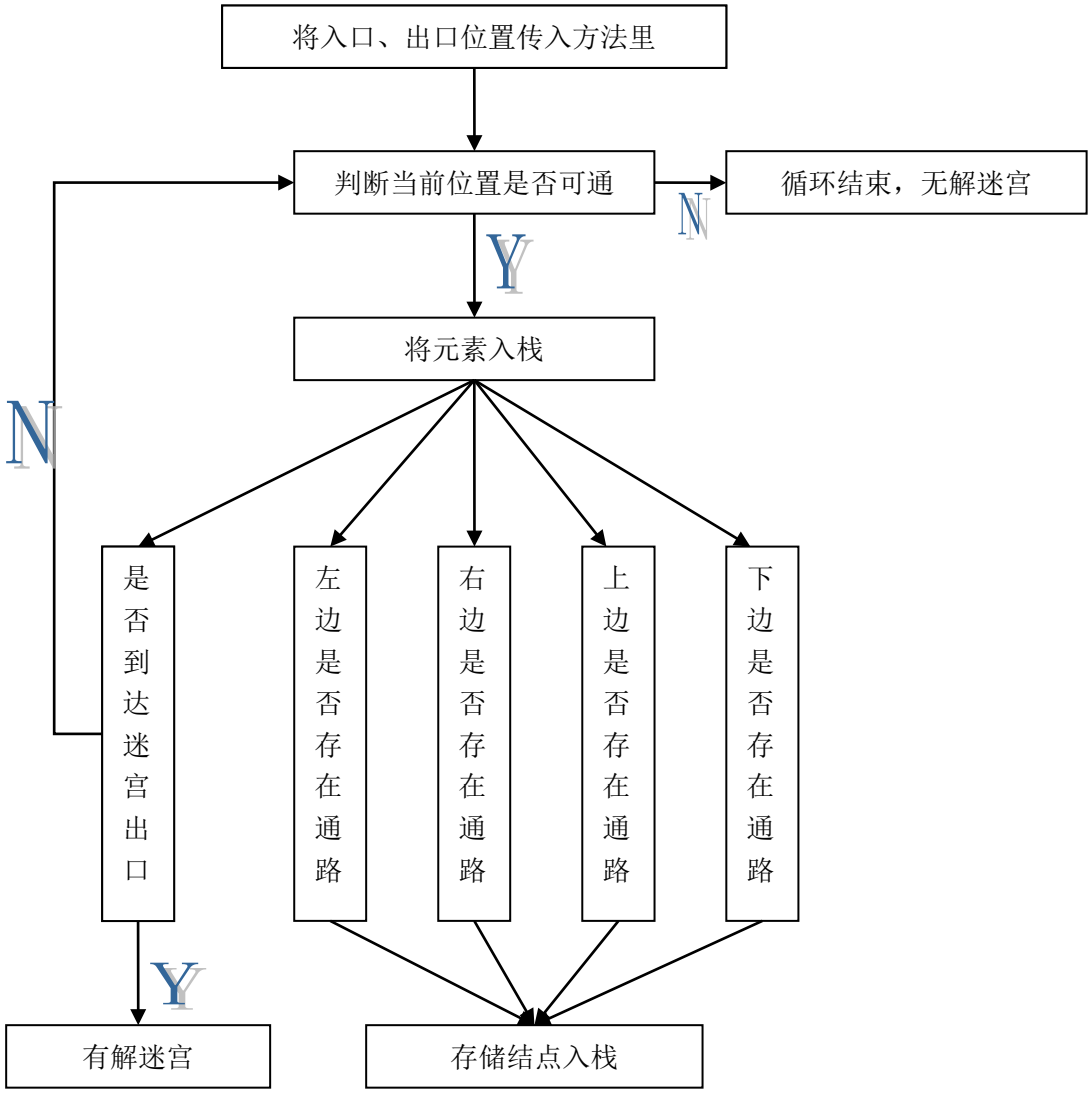


图 3—2

4 软件测试

4.1 调试过程中遇到的问题的解决,以及程序设计思想的实现

（1）调试过程中出现的错误有编译连接时的一些语法错误,也有由于算法存在问题而导致程序运行没有结果或者不是预期的结果。这里主要说一下自己由于算法而导致的错误：因为没有注意结点不能重复走,所以造成了死循环，程序没有结果.通过分析后把走过的结点变为不可走的结点，这样就避免了重复走到该结点.还有就是判断迷宫是否有通路的这个结果不能如预期的结果那样：有通路就输出所有的通路，没有就输出“没有路径可走!”.因为没有通路可走的时候栈是空的,但是输出所有的路径以后栈也是空的.当迷宫有通路的时候,在输出所有路径以后还是会输出“没有路径可走!”这句话。当迷宫没有通路的时候，输出“没有路径可走！”这句话。因为算法的思想是这样的，所以达不到预期的结果。解决的方法是如果迷宫有通路则输出所有的通路，如果迷宫没有通路，则没有结果输出.

(2)迷宫的求解一般采用的是“穷举求解”,利用栈的思想，先将入口位置进栈,判断方向，若其方向可通,则进栈,若不通，则出栈，如此循环下去.

4.2 测试数据

迷宫输入