**武汉大 学计算机学院**

**本科生实验报告**

**数据结构实验报告**

**实验八：利用图搜索方法来解决迷宫问题**

专 业 名 称 ：计算机科学与技术

课 程 名 称 ：数据结构

指 导 教 师 ：安 扬

学 生 学 号 ：2017301500061

学 生 姓 名 ：彭 思 翔

学 生 班 级 ：计科二班

上 机 环 境 ：Visual Studio Code

二○一八 年 12 月

**一、实验题目**

实验八：利用图搜索方法来解决迷宫问题

【问题描述】解决迷宫问题。

【基本要求】

设计程序建立迷宫对应的邻接表表示；

采用深度优先遍历算法输出从入口（1，1）到出口（M，N）的所有迷宫路径。

**二、实验目的**

深入掌握DFS遍历图。

1. **实验项目程序结构**

add

init

main

dfs

**四、实验项目中各文件函数功能描述**

void init(); //读入图

void dfs(int); //深搜迷宫解

void add(int, int); //将两点之间添加一条边

**五、算法描述**

【数据结构】

迷宫：迷宫的每个点(x,y)对应的编号pos为x\*m+y，用以位置编号为下标的head数组存储临边指针，指向的数据为指向的编号和下一个临边的结构体，属于邻接表。

typedef struct ANode {

int d;

struct ANode \* next;

} ArcNode;

ArcNode\* head[MaxN \* MaxN];

【设计思路】

读入迷宫：对每个点，在当前点(i,j)不是墙的情况下，上面的点(i-1,j)不是墙时，将(i,j)和(i-1,j)连接无向边（即两边可互达）,左面的点(i,j-1)不是墙时，将(i,j)和(i,j-1)连接无向边（即两边可互达）。读入的图可不必存下来，只需保存上一行的信息即可。

时间复杂度O(n\*m)

void init() {

scanf("%d%d", &n, &m);

int p[m] = {0};

for (int i = 0; i <= pos(n + 1, m + 1, m); i++) head[i] = NULL;

for (int j = 0; j <= m + 1; j++) p[j] = 1;

for (int i = 1; i <= n; i++)

for (int j = 1; j <= m; j++) {

int key = 1;

if (p[j] == 0) key = 0;

scanf("%d", &p[j]);

if (key == 0 && p[j] == 0)

add(pos(i - 1, j, m), pos(i, j, m)),

add(pos(i, j, m), pos(i - 1, j, m));

if (p[j - 1] == 0 && p[j] == 0)

add(pos(i, j - 1, m), pos(i, j, m)),

add(pos(i, j, m), pos(i, j - 1, m));

}

}

搜索迷宫解：用visit数组标记该编号是否访问过，防止重复访问。直接用邻接表枚举临边挨个访问即可。

输出迷宫解：用一个栈stack记录访问的节点，访问则入栈，退出访问则出栈，这样找到终点时栈内存放的是按解的路径顺序的位置编号。

时间复杂度最多O(n\*m)

void dfs(int x) {

stack[top++] = x;

if (x == pos(n, m, m)) {

ans += 1;

for (int i = 0; i < top; i++) {

printf("(%d,%d)", stack[i]/m, stack[i]%m);

if (i != top - 1) printf(" -> ");

else puts("");

}

top--;

return;

}

visited[x] = 1;

for (ArcNode \*nx = head[x]; nx != NULL; nx = nx->next)

if (!visited[nx->d])

dfs(nx->d);

visited[x] = 0;

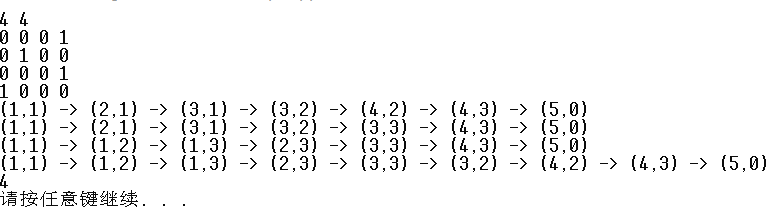
top--;

return;

}

**六、实验数据和实验结果分析**

运行结果良好。



**七、实验体会**

这次图的实验太水了，应该弄一些复杂的图论算法和数据结构。这完全是在复习简单的深搜根本没有到什么图论。我觉得最短路和生成树等算法更合适。即使这样我还是在输入上尝试对空间做了优化。