

《数据结构》实验报告

姓名： 吴泽同 学号： 084622109
班级： 计算机 221 日期： 2023/10/20
程序名： 数据结构实验 7：图的构建与遍历

一、上机实验的问题和要求：

1、图的邻接矩阵存储与遍历（必做题）

(1) 采用邻接矩阵存储图建图（包括有向图和无向图）

(2) 实现图的深度优先和广度优先遍历

输入：图的顶点个数、顶点信息、边的数量、每条边的邻接顶点，如

4

A B C D //4 个顶点，分别是 A、B、C、D

3

A B

A C

B D //3 条边，分别是 (A, B)、(A, C)、(B, D)

2、图的邻接表存储与遍历（必做题）

(1) 采用邻接表存储图建图（包括有向图和无向图）

(2) 实现图的深度优先和广度优先遍历

(3) 输入：图的顶点个数、顶点信息、边的数量、每条边的邻接顶点，如

3、图的应用（选做题，程序设计竞赛的热门考点，不需要写实验报告）

二、程序设计的基本思想，原理和算法描述：

（包括程序的结构，数据结构，输入/输出设计，符号名说明等）

1. 邻接矩阵存储与遍历：

输入：顶点个数 n ，顶点信息 $vertex$ ，边的数量 m ，每条边的邻接顶点 $edges$

创建一个 $n \times n$ 的二维数组 $matrix$ ，所有元素初始化为 0

对于每条边 $(v1, v2)$ in $edges$ ：

找到 $v1$ 和 $v2$ 在 $vertex$ 中的索引 i 和 j

$matrix[i][j] = 1$

如果是无向图，那么 $matrix[j][i] = 1$

深度优先遍历：

创建一个布尔数组 $visited$ ，所有元素初始化为 $false$

从第一个顶点开始，调用 DFS 函数

DFS 函数（顶点 v ）：

标记 v 为已访问

对于 v 的每个邻接顶点 u ：

如果 u 未被访问，那么调用 DFS 函数 (u)

广度优先遍历：

创建一个布尔数组 `visited`，所有元素初始化为 `false`
 创建一个队列 `q`
 将第一个顶点加入 `q`，并标记为已访问
 当 `q` 不为空时：
 出队一个顶点 `v`
 对于 `v` 的每个邻接顶点 `u`：
 如果 `u` 未被访问，那么将 `u` 加入 `q`，并标记为已访问
 2. 邻接表存储与遍历：
 输入：顶点个数 `n`，顶点信息 `vertex`，边的数量 `m`，每条边的邻接顶点 `edges`
 创建一个大小为 `n` 的数组 `list`，每个元素都是一个空列表
 对于每条边 `(v1, v2)` in `edges`：
 找到 `v1` 和 `v2` 在 `vertex` 中的索引 `i` 和 `j`
 将 `j` 添加到 `list[i]` 中
 如果是无向图，那么将 `i` 添加到 `list[j]` 中
 深度优先遍历：
 创建一个布尔数组 `visited`，所有元素初始化为 `false`
 从第一个顶点开始，调用 `DFS` 函数
`DFS` 函数（顶点 `v`）：
 标记 `v` 为已访问
 对于 `v` 在 `list` 中的每个邻接顶点 `u`：
 如果 `u` 未被访问，那么调用 `DFS` 函数（`u`）
 广度优先遍历：
 创建一个布尔数组 `visited`，所有元素初始化为 `false`
 创建一个队列 `q`
 将第一个顶点加入 `q`，并标记为已访问
 当 `q` 不为空时：
 出队一个顶点 `v`
 对于 `v` 在 `list` 中的每个邻接顶点 `u`：
 如果 `u` 未被访问，那么将 `u` 加入 `q`，并标记为已访问

三、源程序及注释（说明每个文件的文件名即可，电子档的

***.h 和 *.cpp 文件压缩传到 FTP 上）:**

实验 7.zip

四、运行输出结果:

（可以将运行结果抓图贴至此处）

输入以课上江苏十三地级市举例为例

```
D:\wu\数据结构实验\6\bin\ x + v
请输入图的顶点个数：13
请输入第1个顶点的信息：南京
请输入第2个顶点的信息：无锡
请输入第3个顶点的信息：徐州
请输入第4个顶点的信息：常州
请输入第5个顶点的信息：苏州
请输入第6个顶点的信息：南通
请输入第7个顶点的信息：连云港
请输入第8个顶点的信息：淮安
请输入第9个顶点的信息：盐城
请输入第10个顶点的信息：扬州
请输入第11个顶点的信息：镇江
请输入第12个顶点的信息：泰州
请输入第13个顶点的信息：宿迁
请输入图的边的数量：12
请输入第1条边依附的两个顶点的编号：南京 常州
请输入第2条边依附的两个顶点的编号：南京 扬州
请输入第3条边依附的两个顶点的编号：南京 镇江
请输入第4条边依附的两个顶点的编号：常州 无锡
请输入第5条边依附的两个顶点的编号：扬州 淮安
请输入第6条边依附的两个顶点的编号：扬州 盐城
请输入第7条边依附的两个顶点的编号：镇江 泰州
请输入第8条边依附的两个顶点的编号：无锡 苏州
请输入第9条边依附的两个顶点的编号：淮安 宿迁
请输入第10条边依附的两个顶点的编号：盐城 连云港
请输入第11条边依附的两个顶点的编号：宿迁 徐州
请输入第12条边依附的两个顶点的编号：苏州 南通
深度优先遍历序列是：
南京常州无锡苏州南通扬州淮安宿迁徐州盐城连云港镇江泰州
广度优先遍历序列是：
南京常州扬州镇江无锡淮安盐城泰州苏州宿迁连云港南通徐州
Process returned -1073740940 (0xC0000374) execution time : 129.051 s
Press any key to continue.
|

D:\wu\数据结构实验\6\bin\ x + v
请输入图的顶点个数：13
请输入每个顶点的信息：南京 无锡 徐州 常州 苏州 南通 连云港 淮安 盐城 扬州 镇江 泰州 宿迁
请输入边的数量：12
请输入每条边依附的两个顶点的编号（下标从0开始）：南京 常州 南京 扬州 南京 镇江 常州 无锡 扬州 淮安 扬州 盐城 镇江 泰州
无锡 苏州 淮安 宿迁 盐城 连云港 宿迁 徐州 苏州 南通
该图是有向图还是无向图？（1表示有向图，0表示无向图）：1
深度优先遍历序列是：南京 镇江 泰州 扬州 盐城 连云港 淮安 宿迁 徐州 常州 无锡 苏州 南通
广度优先遍历序列是：南京 镇江 扬州 常州 泰州 盐城 淮安 无锡 连云港 宿迁 苏州 徐州 南通
Process returned 0 (0x0) execution time : 25.910 s
Press any key to continue.
|
```

五、调试和运行程序过程中产生的问题及采取的措施：

在实际编程过程中，可能会遇到各种问题，例如数组越界，空指针异常，递归深度过大等。对于这些问题，我们需要使用调试工具来定位问题，然后修改代码。例如，如果遇到数组越界，我们需要检查我们的索引是否正确；如果遇到空指针异常，我们需要检查我们是否正确初始化了所有的对象。

六、对算法的程序的讨论、分析，改进设想，其它经验教训：

这两个算法的时间复杂度都是 $O(n^2)$ ，其中 n 是顶点的数量。如果图的边数远小于顶点数的平方，那么使用邻接表可能会更有效率。此外，DFS 和 BFS 都有各自的用途，DFS 可以用来找到所有的连通分量，而 BFS 可以用来找到从一个顶点到另一个顶点的最短路径。

七、对数据结构教学的意见和建议：

暂时没有建议