**\_\_\_\_2019\_\_\_\_\_\_级\_\_\_\_\_信1901-1\_\_\_\_\_班\_\_\_\_\_2020\_\_年\_\_\_\_11\_\_月\_\_20\_\_日**

**姓名\_\_\_\_\_李朴凡\_\_\_\_\_\_\_ 学号\_\_\_\_\_\_20194311\_\_\_\_\_\_ 电话\_\_\_\_\_18333135009\_\_\_\_\_\_\_\_**

**1.实验题目**

图是应用极为广泛的数据结构，也是这门课程的重点。它的特点在于非线性。稀疏矩阵 的十字链表存储结构也是图的一种存储结构，故也把它们归在这次实习中。本章实习继续突 出了数据结构加操作的程序设计观点，但根据这两种结构的非线性特点，将操作进一步集中 在遍历操作上，因为遍历操作是其他众多操作的基础。遍历逻辑的（或符号形式的）结构， 访问动作可是任何操作。本次实习还希望达到熟悉各种存储结构的特征，以及如何应用树和 图结构解决具体问题（即原理与应用的结合）等目的。

**2．需求分析**

分别用邻接矩阵和邻接表实现以下操作：图的创建、遍历、插入、删除、最短路径。

顶点信息边信息的存储，数据结构类型

图的操作 查找，邻接表到有向图

路径输出。判断是否为通路。

图的深度遍历。

**3．概要设计**

设图的结点不超过 30 个，每个结点用一个编号表示（如果一个图有 n 个结点，则它们 的编号分别为 1,2,…,n）。通过输入图的全部边输入一个图，每个边为一个数对，可以对边 的输入顺序作出某种限制。注意，生成树的边是有向边，端点顺序不能颠倒。

（1）借助于栈类型（自己定义和实现）将深度优先遍历用非递归算法实现。

（2）以邻接表为存储结构建立深度优先生成树和广度优先生成树，再按凹入表或树形打印生成树。

**4．详细设计**

数据储存方式

typedef struct Ver {//顶点信息

char num[5];

char name[51];

char instruct[101];

}Ver;

typedef struct {//邻接矩阵

Ver vex[Max];//顶点表

int arcs[Max][Max];

int vnum, arcnum;

}AMGragh;

菜单栏

void menu() {

cout << " 欢迎您" << endl;

cout << " 1、查看所有景点 " << endl;

cout << " 2、景点查询 " << endl;

cout << " 3、最短路径 " << endl;

cout << " 4、修改景点基本信息 " << endl;

cout << " 5、退出 " << endl;

cout << " " << endl;

cout << "请选择";

}

**5．调试分析**

不能重复保留上一个问题，对于最短路径问题是不完全符合实际。所有景点是前期数据保存的，不需要人为的输入。

**6．使用说明**

根据菜单栏提示选择要执行程序即可。

**7．测试结果**

