2018/11/28

确定了一些操作的步骤：

广度优先搜索BFS

基本实现思想：

（1）顶点v入队列。

（2）当队列非空时则继续执行，否则算法结束。

（3）出队列取得队头顶点v；

（4）查找顶点v的所以子节点，并依次进入队列；

（5）转到步骤（2）。

深度优先搜索DFS

算法基本思想：

（1）访问顶点v，打印节点；

（2）遍历v的子节点w，while（w存在），递归执行该节点；

2018/11/29

完成了框架的搭建

图的节点：

struct Edge {

int vertex;

int wt;

Edge();

Edge(int, int);

};

struct Vertex {

int vertex;

std::vector<Edge>\* edge;

};

图的ADT：

class Graph {

public:

Graph() = default;

~Graph() = default;

virtual void Init(int) = 0;

virtual int n() = 0;

virtual int e() = 0;

virtual int getFirst(int) = 0;

virtual int next(int, int) = 0;

virtual void setEdge(int, int, int) = 0;

virtual void deleteEdge(int, int) = 0;

virtual bool isEdge(int, int) = 0;

virtual int getWeight(int, int) = 0;

virtual int getMark(int) = 0;

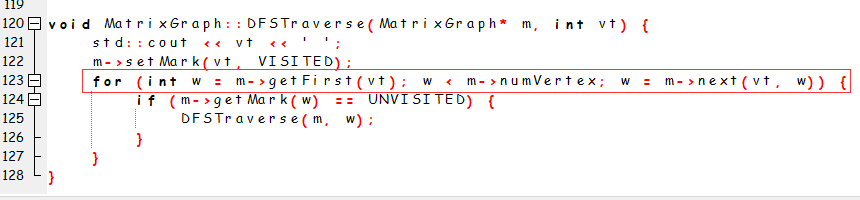
virtual void setMark(int, int) = 0;

virtual void printGraph() = 0;

};

2018/11/10 20：16

在实现DFS操作的时候遇到了问题，由于不能保证图是连通的，导致与进行搜索的节点不相连的子图无法被访问，错误如下：



加入循环代码后即可保证每个节点都是被访问过的

2018/11/30

在实现邻接矩阵表示图的过程中总结了一下几点规律

* 无向图的邻接矩阵都是沿对角线对称的
* 要知道无向图中某个顶点的度，其实就是这个顶点vi在邻接矩阵中第i行或（第i列）的元素之和；
* 对于有向图，要知道某个顶点的出度，其实就是这个顶点vi在邻接矩阵中第i行的元素之和，如果要知道某个顶点的入度，那就是第i列的元素之和。