哈尔滨工业大学

**<<数据结构与算法>>**

**实验报告**

**(2020年秋季学期)**

|  |  |
| --- | --- |
| **姓名：** | **姚舜宇** |
| **学号：** | **1190202107** |
| **学院：** | **计算机学院** |
| **教师：** | **赵妍妍** |

实验三 图

## 一、实验内容

1. 输入节点数量，边的数量，以及每条边所连接的节点和边权，构造无向带权图。
2. 实现顶点的增加、删除、查找。

增加：输入一个顶点编号，将该节点加入图中。

删除：输入一个顶点编号，在图中删除该顶点，以及与其相连的所有边。

查找：输入一个顶点编号，输出该顶点是否存在，哪些顶点与其直接相连。

1. 实现边的增加、删除。

增加：输入两个顶点编号，在图中添加一条边连接这两个顶点。需判断顶点是否存在。

删除：输入两个顶点编号，删除图中所有直接连接这两个顶点的边。如不存在这样的边则不处理。

1. 从一个给定顶点开始，实现图的深度优先遍历（此时不考虑边权）。
2. 从一个给定顶点开始，实现图的广度优先遍历（此时不考虑边权）。
3. 计算给定顶点的到其他顶点的最短路径(要求使用Dijkstra算法)。
4. 计算图中任意两顶点之间的最短路径(要求使用Floyd算法)。

## 二、实验过程及结果

使用邻接表存储图。

初始化图：输入顶点，边的数目，依次输入每一条边依附的两个顶点和权值。使用一个Vertex型的vector容器，每个Vertex包括顶点编号，此顶点邻接的一条边。

增加顶点：输入顶点的编号后，新建一个Vertex，加入到vector中。

删除顶点：输入顶点的编号后，在vector内遍历每个顶点的每一条边，如果此边有某一个依附的顶点是要删除的顶点，删除该边。遍历到要删除的顶点时，直接从vector中将该顶点删除。

查找顶点：输入顶点的编号后，在vector内遍历每个顶点，如果查询到，则遍历其邻接的边，输入依附的令一个顶点。如果没有查询到，则输出提示表示没有查询到。

增加边：输入边依附的两个顶点和权值，在这两个Vertex的表尾各自新插入一个Edge即可。如果顶点不存在，则输出提示表示没有查询到。

删除边：输入边依附的两个顶点，如果存在，则分别删除两个Vertex连接的边即可。如果不存在，则输出提示表示没有查询到。

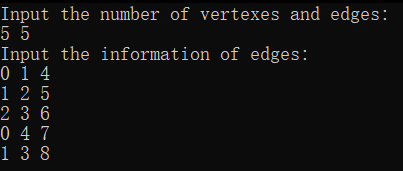
深度优先搜索：遍历每个顶点，标记是否访问过，用递归进行。

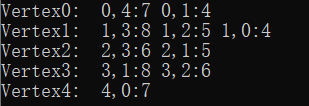
广度优先搜索：使用队列，标记是否访问过，将每个顶点依次放入队列中，每次弹出头节点，并将其邻接的且未访问过的顶点放入队列中，循环至队列空。

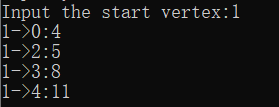
给定顶点的到其他顶点的最短路径：使用Dijkstra算法，需要一个存储已经确定到源点最短路径的集合，和存储源点到其余顶点最短路径的数组。在未确定最短路径的顶点集合里寻找使得此顶点到源点距离最小的顶点，加入集合中，如果源点到其余顶点的距离大于中间经过新加入的顶点所确定的距离，则更新存储最短路径的数组。循环直到存储已经确定到源点最短路径的集合等于图的顶点集合。

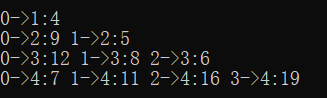
任意两顶点之间的最短路径：使用Floyd算法，构造邻接矩阵，对图的顶点三重循环，如果邻接矩阵中某两点之间的距离大于中间经过某一顶点所确定的距离，则更新邻接矩阵。

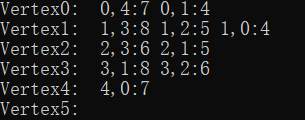
测试用例：

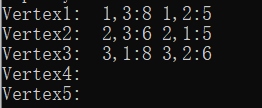


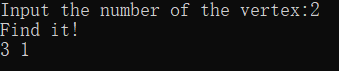


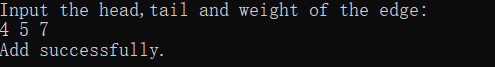
Dijkstra: 

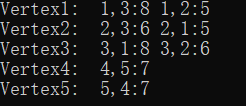
Floyd: 

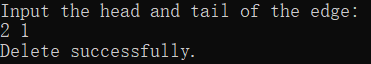
增加顶点：

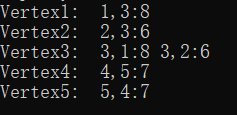
删除顶点：

查找顶点：

增加边：



删除边：



DFS：

BFS：

## 三、实验心得

第一题主要是关于图的基础操作，完成本次实验后，感到使用邻接表存储图，增加、删除点、边，对链表的操作更加熟练；熟悉了对图的遍历，Dijkstra算法，Floyd算法。

遇到的困难：1.某些操作使用邻接表较为麻烦，操作的复杂度较高，不如使用邻接矩阵。2.顶点的编号和序号不一定相等，如第三个顶点的编号为5，故删除边的时候需要将编号和序号进行转换。