## 实验6 图操作及应用

1. 实验目的

（1）理解图的邻接矩阵或邻接表表示方法，能将图抽象为合适的存储模型；

（2）能使用最短路径Dijkstra算法解决校园导航问题。

2. 实验内容

（1）第一关：能实现图的邻接矩阵存储；

（2）第二关：能输入地图，设计实现最短路径Dijkstra算法。

详见educoder

##### 1、数据结构设计

**图的存储**：用邻接矩阵，这样会方便不少。

邻接矩阵是一个二维数组，数组中的元素是边的权（一些数值），数组下标号为结点的标号。

（1）例如二维数组中的一个元素M[5][6]的值为39，则表示结点5、6连接，且其上的权值为39。

（2）用邻接矩阵存储图，对图的读写就简单了。因为邻接矩阵就是一个二维数组，因此对图的读写就是对二维数组的操作。只要能弄清楚边的编号，就能把图读入了。

用一对结点表示边（也就是输入的时候输入一对结点的编号）

##### 2、实验要求算法

1.Dijkstra生成最短路径

书上给出了完整的求最短路径的源代码。总的来说的不断添加点的过程，把距离原点路径最短的点添加（通过做标记的方法）并更新没有做标记的图中点距离原点的距离，直到所有的点都做了标记。

自己编写代码的时候注意，书上仅仅是给出了最短路径的源代码，离一个完整的程序还有距离，其中要补充图的读写的函数和定义一些常量。关键是要明白书上代码的意思。

求最短路径的Dijkstra算法：

求最短路径就是求图中的每一个点到图中某一个给定点（这里认为是编号为0的点）的最短距离。

具体算法就是初始有一个旧图，一个新图。开始的时候旧图中有所有的结点，新图中初始为只有一个结点（源点，路径的源头）。整个算法就是不停的从旧图中往新图中添加点，直到所有的点都添加到新图中为止。

要实现这个算法，除了用二维数组保存图，还需要使用到两个辅助的数组

数组find[N]：此数组是用来表示标号对应的结点是否已经被添加到新图中（因为只有旧图中的点我们才需要添加到新图中，并且只有旧图中点到源点的距离，我们才需要进行更新）其中N为图中结点的个数。

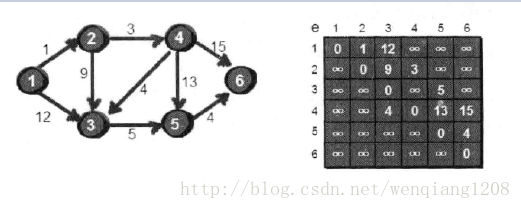
数组distance[N]：此数组记录图中的点到源点的距离。这个数组里面存放的值是不断进行更新的。其中N为图中结点的个数。

大的循环：逐一把旧图中的所有点添加到新图中。

把找到的结点添加到新图中。

循环：遍历distance数组，找distance数组中属于旧图中的点，其中distance最小的那个结点

更新distance数组，因为新添加了一个点到新图中，所以可能能找到一条通过这条点的新的路径，这条路径可能比原来的路径更短。



计算结果为：



下面的测试用图见上面的图片中的图。 测试输入（前面9行都是构造图需要的起点 终点 权值，最后一行是须计算的路径的起点和终点）：

1 2 1

1 3 12

2 3 9

2 4 3

4 3 4

4 5 13

3 5 5

4 6 15

5 6 4

0 6

预期输出（输出的是0号点到6号点的最短路径长度）：

17

测试输入：

1 2 1

1 3 12

2 3 9

2 4 3

4 3 4

4 5 13

3 5 5

4 6 15

5 6 4

0 3

预期输出： 8

要求:

使用文件输入数据

使用文件保存结果

第十三周验收