实验七 行车路线

一、问题分析

1. 分析并确定要处理的对象（数据）是什么

问题描述

小明和小芳出去乡村玩，小明负责开车，小芳来导航。小芳将可能的道路分为大道和小道。大道比较好走，每走 1 公里小明会增加 1 的疲劳度。小道不好走，如果走小道，小明的疲劳值会快速增加，走 s 公里小明会增加 s^2 的疲劳度。所有的小道不相交。例如：有 5 个路口，1 号路口到 2 号路口为小道，2 号路口到 3 号路口为大道，3 号路口到 4 号路口为大道，4 号路口到 5 号路口为小道，相邻路口之间的距离都是 2 公里。如果小明从 1 号路口到 5 号路口，则总疲劳值为 $2^2+2+2+2^2=4+2+2+4=12$。现在小芳拿到了地图，请帮助她规划一个开车的路线，使得按这个路线开车小明的疲劳度最小。

问题求解

问题是一个行车路线的规划乡村里面有若干个路口和若干条道路，不同的道路带给驾驶员的疲劳度（每公里）不一样，要求我们设计一个最优的行车路线，使得驾驶的疲劳度最小。因为路口之间的道路有着不同的连接关系，所以对应着一种非线性的多对多的关系，可以用图来处理这个问题。又由于道路之间相互连通，故图为无向图。处理的对象就是一个无向图，赋给道路不同的权值，这个权值对应的就是疲劳度。

2. 分析并确定要实现的功能是什么

由上述分析可知，处理的对象是一个无向图中从顶点 1 到顶点 n 的最短路径，路径的权值根据道路来确定，如果是大道的话，权值为原来道路的长度，如果是小道的话，权值为原来道路长度的平方，实现求出 1 到 n 中所有路径中最短路径的长度。

3. 分析并确定处理后的结果如何显示

利用一个数组去存储从第一个路口到第 i 个路口的最短距离（i=2,3,…..n）,根据题目要求求出 1 号路口到 n 号路口的最小“路径”即输出数组最后一个值，就是所求的最优路线下的最小疲劳度。

二、数据结构和算法设计

1. 抽象数据类型设计

考虑到用 Dijkstra 算法来求解图中各点到第一个点之间的最短路径，由于道路之间是双向道路，故采用邻接矩阵来处理图较好。矩阵中每个点的坐标表示各点的序号，矩阵中点的值表示每公里驾驶增加的疲劳度（这里可以看做是图的权值）。ADT 中还需要有一个标记数组来标记顶点，一个顶点数组去存储顶点信息，一个 bool 型变量去判断图的类型，两个变量 numVert,numEdge 分别表示顶点个数和图之间的边数。

基本操作需要实现对图的初始化，返回顶点个数，边数，在存储图的时候需要在邻接矩阵中设置顶点信息和边权，用 setVex,setEdge 两个函数来实现。

2. 物理数据对象设计（不用给出基本操作的实现）

无向图的存储用一个邻接矩阵来进行存储，在输入长度的时候要根据道路的类型来确定相应的权值。邻接矩阵的实现用一个二维指针，实质上就是一个二维数组表，表中的内容就是相应的权值。

到各个点之间的最短距离用一个数组来存储，每次求出到第 i 个点的更小的距离时，就更新这个距离值，最后遍历完所有点，可以得到存储各个点到第一个点的最短距离

3. 算法思想的设计

算法思想：用 Dijkstra 算法来求解图的最短路径。一般图中无非负权值的时候，解决单源点最短路问题，可以采用 Dijkstra 算法。首先初始化最短距离的数组 d，遍历一个点 w，找到与这个点相连的点，比较 d[w]和 d[v]加上权值的和的大小关系，更新 d[w]为较小的值，遍历完所有结点就可以得到从结点 1 到结点 n 的最短路径。

4. 请用题目中样例，基于所设计的算法，详细给出样例求解过程。

输入样例为

6 7
1 1 2 3
0 2 3 2
0 1 3 30
0 3 4 20
0 4 5 30
1 3 5 6
0 5 6 1

将其装换为用邻接矩阵表示出图

0 9 30 0 0 0
9 0 2 0 0 0
30 2 0 20 36 0
0 0 20 0 30 0
0 0 36 30 0 1
0 0 0 0 1 0

定义一个最短距离的数组 d[n],并进行初始化。

第一次进行计算得到的数组 d[n]

顶点	1	2	3	4	5	6
d[i]	0	9	30	INFINITY	INFINITY	INFINITY

第二次进行计算得到的数组 d[n]

顶点	1	2	3	4	5	6
d[i]	0	9	11	INFINITY	INFINITY	INFINITY

第三次进行计算得到的数组 d[n]

顶点	1	2	3	4	5	6
d[i]	0	9	11	31	47	INFINITY

第四次进行计算得到的数组 d[n]

顶点	1	2	3	4	5	6
d[i]	0	9	11	31	47	INFINITY

第五次进行计算得到的数组 d[n]

顶点	1	2	3	4	5	6

d[i]	0	9	11	31	47	48

最后由表可知从路口 1 到 6 的最短路径为 48.

5. 关键功能的算法步骤（不能用源码）

Dijkstra 算法步骤

一般图中无非负权值的时候，解决单源点最短路问题，可以采用 Dijkstra 算法。

首先初始化最短距离的数组 d，遍历一个点 w，找到与这个点相连的点，比较 d[w]和 d[v]加上权值的和的大小关系，更新 d[w]为较小的值，遍历完所有结点就可以得到从结点 1 到结点 n 的最短路径。

三、算法性能分析

算法复杂度的分析

时间复杂度的分析：主程序中对图进行初始化，用一个循环设置顶点信息，时间复杂度为 O（n）,对输入图的边权进行初始化，时间复杂度为 O（m）(m 为边数)，用一个循环初始化最短距离的数组 d[]，时间复杂度为 O（n）,Dijkstra 算法中遍历整个邻接矩阵，用二重循环实现，内部循环有一个对最短距离是否更新的判断语句，整个 Dijkstra 函数的时间复杂度为 O（n^2），由化简规则可知算法的时间复杂度为 O（n^2）。

空间复杂度的分析：初始化邻接矩阵，动态开辟一个大小为 N 的标记数组，其余操作均无动态内存创建，故空间复杂度为 O（n）.