## 图论实验二 制作文档

一、实验内容

从文件input.txt中读入一个有向正权图的权矩阵表示，程序按要求通过Dijkstra算法输出这个图中某一结点到其余各结点的最短路径长度。本程序以Windows平台下的Visual Studio 2012为开发环境，用C++语言实现上述功能。

二、实验要求

输入数据保存在文件input.txt 中，程序运行后将输出数据保存至**[**学号**].txt**中。这两个文件都与程序在相同的路径下。 输入文件第一行为一个整数n，表示图中结点的个数；输入文件第一行为一个整数n，表示图中结点的个数；接下来有n行，每一行有n个用空格隔开的整数，第i行的第j个整数表示矩阵中的aij；第n+2行是一个整数k（1≤k≤n），表示需要计算的最短路径的起始结点编号。

输出文件仅有一行，包含n-1个由空格隔开的正整数，分别表示第k个结点到其余各结点的最短路径长度，按照结点编号由小到大排列。特别的，如果两个结点之间不连通则输出-1。

三、设计思路

鉴于输入图的规模从1至31个节点不等，程序通过new生成动态二维数组，动态数组等元素以存储权矩阵，至各节点路径值以及节点是否已找到最短路径等。

程序首先读入文件中的节点个数v，随后用其生成上述动态结构，将文件中的权矩阵读入QJ[i][j]中，并将起始节点的编号读入变量node当中。通过v生成存储最短路径值以及节点最短路径搜索情况的数组PathCost[i]（即Dijkstra算法中的*π*(i)）与S[i]，其中S[i]值为1的节点代表已找到最短路径，否则值为零。随后根据Dijkstra算法初始化PathCost[i]与S[i]。随后开始一个永真循环，首先通过函数FindMin找出未找到最短路径节点中PathCost最小的节点的编号j，并完成数组S[i]的更新，然后判断S[i]中元素总和是否为节点数，即判断未找到最短路径的节点的集合是否为空，是则break，否则遍历未找到最短路径节点集与j的后继节点集的交集，并更新其中节点的PathCost。最后部分为文件输出，注意不输出起始节点的PathCost，并且PathCost大于100则输出-1即可。

四、关键代码分析

第一个值得注意的代码段是第10行对文件打开是否成功的判断与处理，如果打开失败，则输出“Could not open the file <input.txt>”并结束程序。第24行判断起始节点编号是否大于总节点数，是则返回错误并结束程序。

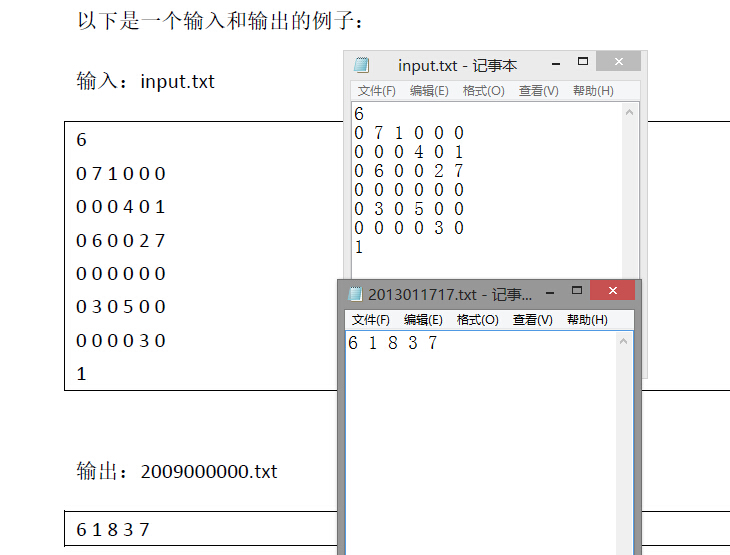
随后根据Dijkstra算法初始化PathCost[i]与S[i]，即若QJ[node][i] != 0，即i为node的后继节点，则PathCost[i] = QJ[node][i]，否则PathCost[i] = 101\*v，意为节点i目前不可到达（利用了题目要求中的边权小于等于100）；对S[i]则赋S[node]为1，其余为0即可。

随后开始循环。函数FindMin找出未找到最短路径节点中PathCost最小的节点的编号j并将j返回，并完成从未找到最短路径节点集中去掉j的操作，即将S[j]赋值为1。随后判断未找到最短路径的节点的集合是否为空，由于可能有无法到达的节点，因此break的条件为S[i]中元素总和没有变化。随后遍历所有节点，通过两个if语句选出未找到最短路径节点集与j的后继节点集的交集中的节点i，将PathCost[i]更改为min(PathCost[i], PathCost[j] + QJ[j][i])，然后开始下一轮循环。

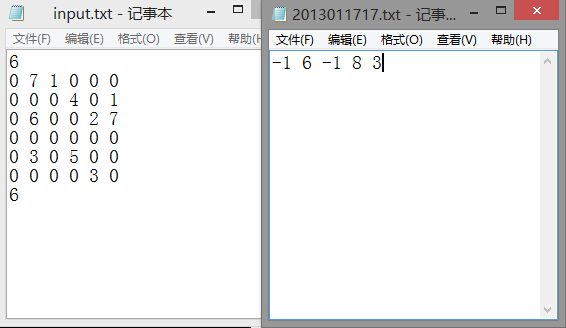
文件输出部分，程序运用C++已有的fstream，用for循环完成PathCost中数据的输出并注意了判断起始节点不输出与PathCost大于100则输出-1的问题以及输出格式的问题，如行末尾没有空格。最后，程序关闭文件input.txt与2013011717.txt。

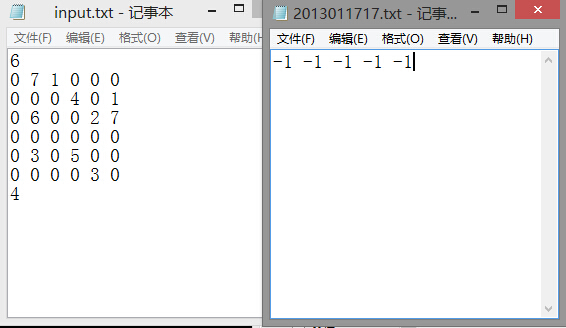
五、实验结果与分析

以下为样例输入与输出：

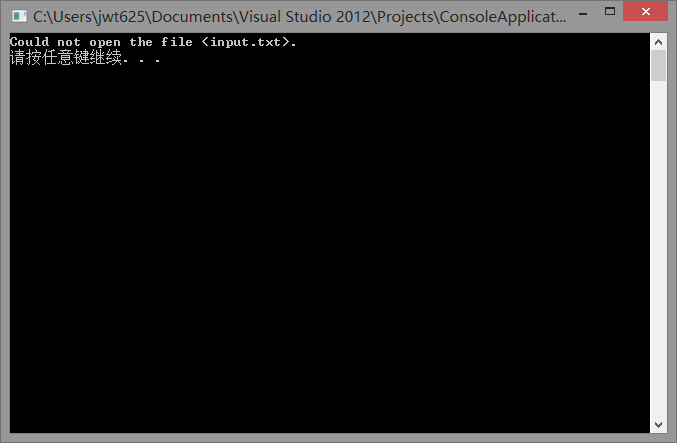


输出与所给输出一致。下面为另两组输入与输出测试：

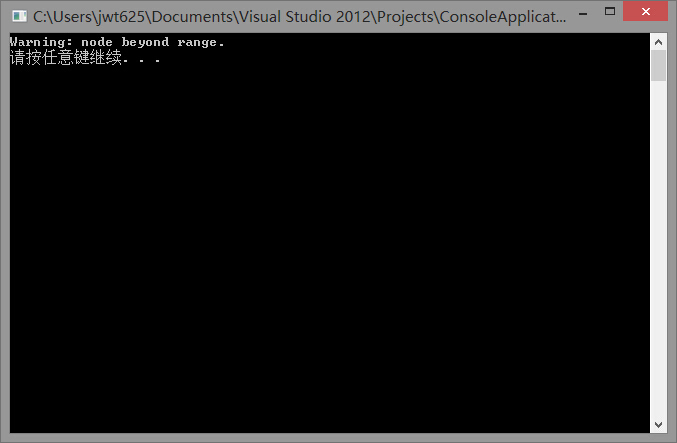




下面为没有input.txt文件时程序的输出：



下面为起始节点编号超出节点总数时的输出：



综上，程序在正确的输入下能够完成正确的输出，并能在读取文件失败与起始节点不合理时反馈给用户。

六、实验小结

本次实验要求从文件读入一个有向正权图的权矩阵表示，程序通过Dijkstra算法输出这个图中某一结点到其余各结点的最短路径长度。通过这次实验，我练习了对Dijkstra算法的具体实现，加深了对Dijkstra算法的理解，以及对其的具体的程序实现进行了实践与熟悉。

不足的是由于编程经验本身较少，程序在不少地方仍可能有所缺陷与漏洞，如未添加判断权矩阵的规模是否与节点数相符；在算法方面也还有许多地方有待改进，如均用动态数组可能减慢运行速度。以上就是本次实验的小结。