**★★★☆☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：11015: 05-2 Rendezvous**

**解題者：林澤丞**

**解題日期：2017年4月6日**

**題意：**

有一個小組要在一個地方討論功課。他們想要在一個可以讓總耗費的量最少的地方見面。總耗費量的定義是從所有人到目的地的總距離。輸入格式：第一行有二個整數N、M，N表示小組人數，M表示路徑數。接下來的N行代表組員的名字。接下來M行有3個整數，i、j、k，表示從i到j或從j到i的路徑需耗費k。如果N、M都是0表示程式結束。

**題意範例：**

Sample Input:

4 3

timotius

harry

richard

januar

1 2 10

1 3 8

1 4 6

Sample Output

Case #1 : timotius

**解法：**

這是All pairs shortest paths problem，可以用dynamic programming的方式解。

求出陣列$A^{k}$後，將每一行的數字加起來。陣列中第*i*行的總和代表所有點到目的地*i*的總耗費。將總消耗最小的編號所對應的名字印出來。

**解法範例：**

路徑及距離如圖所示，求All pairs shortest paths。****

$$A^{0} =   \begin{matrix}0&8&\infty &1\\8&0&3&\infty \\\infty &3&0&2\\1&\infty &2&0\end{matrix}$$

$$A^{1}= \begin{matrix}0&8&\infty &1\\8&0&3&9\\\infty &3&0&2\\1&9&2&0\end{matrix}$$

$$A^{2}=\begin{matrix}0&8&11&1\\8&0&3&9\\11&3&0&2\\1&9&2&0\end{matrix}$$

$$A^{3}=\begin{matrix}0&8&11&1\\8&0&3&5\\11&3&0&2\\1&5&2&0\end{matrix}$$

$$A^{4}=\begin{matrix}0&6&3&1\\6&0&3&5\\3&3&0&2\\1&5&2&0\end{matrix}$$

依據$A^{4}$可以得到：

所有點到目的地1的總耗費為：$0+6+3+1=10$

所有點到目的地2的總耗費為：$6+0+3+5=14$

所有點到目的地3的總耗費為：$3+3+0+2=8$

所有點到目的地4的總耗費為：$1+5+2+1=8$

**討論：**

無

**程式：**

#include<stdio.h>

#define MAX\_DISTANCE 100000

#define MAX\_CASE 22

int main()

{

 int n,m;

 int i,j,k;

 char name[MAX\_CASE][11];

 int distance[MAX\_CASE][MAX\_CASE];

 int distance\_to\_place[MAX\_CASE];

 int place1,place2,cost;

 int case\_number=0;

 while(1)

 {

 scanf("%d %d",&n,&m);

 if(n==0&&m==0)

 break;

 for(i=0;i<n;i++)

 {

 scanf("%s",name[i]);

 }

 for(i=0;i<n;i++)

 {

 for(j=0;j<n;j++)

 {

 if(i==j)

 distance[i][j]=0;

 else

 distance[i][j]=MAX\_DISTANCE;

 }

 }

 for(i=0;i<m;i++)

 {

 scanf("%d %d %d",&place1,&place2,&cost);

 distance[place1-1][place2-1]=cost;

 distance[place2-1][place1-1]=cost;

 }

 for(k=0;k<n;k++)

 {

 for(i=0;i<n;i++)

 {

 for(j=0;j<n;j++)

 {

 if(distance[i][k]+distance[k][j]<distance[i][j])

 {

 distance[i][j]=distance[i][k]+distance[k][j];

 }

 }

 }

 printf("k=%d\n",k);

 for(i=0;i<n;i++)

 {

 for(j=0;j<n;j++)

 {

 printf("%d ",distance[i][j]);

 }

 printf("\n");

 }

 }

 for(i=0;i<n;i++)

 {

 distance\_to\_place[i]=0;

 }

 for(i=0;i<n;i++)

 {

 for(j=0;j<n;j++)

 {

 distance\_to\_place[i]+=distance[j][i];

 }

 }

 int shortest\_distance=MAX\_DISTANCE;

 int shortest\_place=100;

 for(i=0;i<n;i++)

 {

 if(distance\_to\_place[i]<shortest\_distance)

 {

 shortest\_distance=distance\_to\_place[i];

 shortest\_place=i;

 }

 }

 printf("Case #%d : %s\n",case\_number+1,name[shortest\_place]);

 case\_number++;

 }

 return 0;

}