ACM Pacific NW Region Programming Contest
10 November 2001

PROBLEM D
DETOURS!

You are working for the Department of Transportation (DOT) in road repair. When a road is closed for repairs, the public would like the marked detour to be the *shortest* available combination of roads to travel from one end of the closed road to the other.

To simplify the data, we will only consider roads between cities/towns/villages/?, rather than trying to give unique identifiers to crossroads that do not have such names.

After your program has read in the road data set, it will read in a series of town name pairs that mark the roads to be closed. Your program is to find the *shortest* detour between each of the two towns, and specify it by listing in order the towns along the detour – you may use either of the two towns as the starting point – and then list the towns along the path to the ending point. It is understood that there is a road between each pair of towns in your listing. At that point, list also the total length of the detour.

Input:

- § The first line contains the number of towns (an integer, *N*) in the data set. The rest of the first line is to be discarded (there may be additional notation text)
- § The next *N* records contain the names of the towns, each on a separate line. They may or may not contain embedded blanks in their names (for instance, `Coeur d'Alene` and `Wilbur`).
- § Subsequent lines contain three items that represent the roads (whitespace delimited):
  1. The town at one end of the road
  2. The town at the other end of the road
  3. The length of the road connecting those two towns.

  If the name of either town contains whitespace, it will be enclosed in double quote marks. (See sample data)
- § The end of road information is marked by this record: `EOD EOD 0`
- § Finally, there are a series of town pairs (whitespace delimited). These each represent a road to be closed for repairs, and consequently, two towns between which you must find the shortest path. Again, should the town names contain whitespace, they will be enclosed in double quotes. This portion of the data set is terminated by this town pair: `EOD EOD`

Input file for this problem will be **D.in**

Output:

- § On one line, list all of the towns along the detour, including both end–point towns. The list can begin with either town.
- § On a separate line following the detour path, give the length of the detour.
- § Follow this with a blank line.
- § Should your program encounter a city that is not in the list of *N* cities, either in the list of roads or in the road–closure pairs, display a single line stating "*CityName* is not a recognized town." If the unrecognized city is the first of that record, do not process the rest of the record.
- § Follow this with a blank line.
- § Should a road–closure pair contain recognized cities, but cities with no direct road between them, display a line stating: "There is no road directly from *CityName1* to *CityName2*." (City names may be in any order in this statement.)
- § Follow this with a blank line.

Sample I/O:

Input:

```
8
Connell
Coulee City
Davenport
George
Moses Lake
Ritzville
Sprague
Wilbur

Connell "Moses Lake"        46
Connell Ritzville           45
Connell Wilbur              99
"Coulee City" George        55
"Coulee City" "Moses Lake"  52
"Coulee City" Wilbur        35
Davenport Sprague           38
Davenport Wilbur            32
George "Moses Lake"         31
"Moses Lake" Ritzville      42
Ritzville Sprague           23

EOD EOD 0


"Coulee City"  "Moses Lake"
George "Moses Lake"
Seattle Spokane

EOD EOD
```

Output:

```
Coulee City George Moses Lake
Total distance: 86 miles

Moses Lake Coulee City George
Total distance: 107 miles

Seattle is not a recognized city.
```