FINAL PROJECT:

Dr. Zoidberg's day out.

Due Date: Thursday, May 12th, 11:59:59pm

For this assignment, you need to submit a file called '`zday.cpp`' and any other necessary implementation files. Remember to put your **name** and **section** at the top of all your files.

Problem:

It is Dr. Zoidberg's day out and he has been lucky. He has found a coupon for a free airplane flight to any city of his choice! After landing in the destination's airport, Dr. Zoidberg can go sightseeing using the city's public transportation tube networks. However, Dr. Zoidberg is short of transportation tube tokens, so he may not be able to pay the fare to reach every location between a city.

Given a map of a city's tube network (given to you as a **Graph**), and how many tube tokens Dr. Zoidberg has, help him figure out which locations he can reach from the city's airport.

The Tube Transportation System.

Input:

The input will consist of a sequence of test cases. Each test case consists of Dr. Zoidberg's starting tokens and a **Weighted Directed Acyclic Graph** that represent the city's tubes and their fares.
The first line of the input tells you how many test cases are in the input.
Each test case starts with a single line <*T*>, Dr. Zoidberg's starting tokens.
For each graph, the first line is of the form:

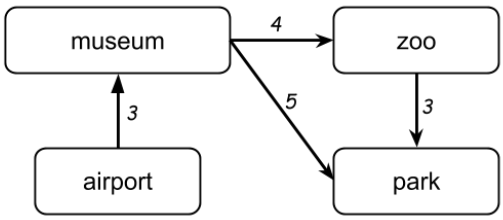
<*name*> <*N*>

where <*name*> is the name of the city and <*N*> is the number of locations in this city.
Afterwards, <*N*> lines follow, each of the form:

<*location*> <*K*> <*loc1*> <*w1*> <*loc2*> <*w2*> …. <*locK*> <*wK*>

where:
<*location*> indicates the name of a location, <*K*> indicates the number of neighbors of <*location*> and <*loc1*><*w1*> <*loc2*><*w2*> …. <*locK*><*wK*> is a list of tubes (edges) out of <*location*> each with destination <*locX*> and a fare price (weight) of <*wX*>.

For example, to represent the graph:



New New York

The input file will read:

```
NewNewYork 4
airport 1 museum 3
museum 2 zoo 4 park 5
zoo 1 park 3
park 0
```

Notice:

1. Location names do not include spaces.
2. Dr. Zoidberg's starting tokens and tube fares are all positive Integers.
3. Every city will include a location called "`airport`". (Not necessarily the first location listed)

Output:

For each city, output which locations Dr. Zoidberg can reach from the "`airport`" given his starting tokens in the format shown in the sample input / output.

Details:

- Build your own graph data-structure. (an “Adjacency Matrix” or an “Adjacency List” as seen in class)

Sample:

Input	Output
3 7 NewNewYork 4 airport 1 museum 3 museum 2 zoo 4 park 5 zoo 1 park 3 park 0 6 Miami 3 airport 2 swamp 7 beach 10 swamp 0 beach 1 swamp 5 15 Rolla 5 airport 2 lake 5 forest 8 lake 1 forest 7 forest 2 river 6 bar 9 river 1 bar 7 bar 0	In NewNewYork, Dr. Zoidberg can reach: museum zoo In Miami, Dr. Zoidberg can reach: nothing :-(In Rolla, Dr. Zoidberg can reach: lake forest river

Useful Hints:

1. Carefully choose how are you going to implement the graph.
2. Write down an algorithm for any function before you start coding it.
3. Develop any member functions one at a time, starting from the simplest ones.
   Move to the next function only after the previous one has been tested.
   Trying to code the whole homework and then remove bugs may prove to be too big a task.

END.