2717 Earthquake Management

An earthquake management software collects and interprets bits of disparate data from emergency calls, news broadcasts, satellite imagery, reading from remote sensors attached to roadways or buildings, estimates the damage caused to road links connecting major towns in a state and produces a report on the condition of road links giving an estimate of the maximum average speed in km/hr (the speed is 0 km/hr if a road link is unusable) with which rescue operations can be executed through each of the existing road links. The information produced by the earthquake management software is to be used to find a set F of fastest independent priority routes from a given set of source towns where adequate resources are available to a given destination town that is affected severely. You are required to write a program to find F.

Assume that towns are identified by integers. If a town p has better resources than another town q then $p < q$. A route from a source to the destination is represented by a sequence of distinct integers representing towns en route. The first integer on a route is a source while the last integer is the destination. If an integer q appears next to an integer p on a route then the road link from town p to town q belongs to the route. A route is associated with the operation time and the covering distance. The operation time of a route is the time required to execute rescue operations at the estimated maximum average speed on each road link on the route. The covering distance of a route is the distance covered to reach the destination from the source following the route.

A fastest route is a route through which rescue operations can be executed in least operation time. If there is more than one fastest route then a fastest route that has shortest covering distance is selected and the route is called a fastest compact route. Again if there are two fastest compact routes then priority is given to the fastest compact route that has better resources en route to the destination. For example if R1 and R2 are two fastest compact routes containing town $p1$, $p2$ ($p1 < p2$) respectively, and town q appears next to $p1$ in R1 and next to $p2$ in R2 then R1 is given priority over R2. The fastest priority route is the fastest compact route with highest priority.

The set F of fastest independent priority routes is defined recursively. The fastest priority route has the priority one to be in F. The fastest independent priority route with priority k is the fastest priority route obtained after removal of all road links appearing in the first $k-1$ fastest independent priority routes.

Input

The input may contain multiple test cases.

For each test case the first input line gives the test case number c. The second input line gives a sequence of integers representing the source towns while the third input line gives an integer representing the destination town.

In addition to the first three input lines mentioned above the input contains a set of lines defining existing road links and their condition after the disaster. The lines are given in an arbitrary order. Each of these lines contains two integers and two real numbers. The integers represent two towns between which there is an existing road link. The first one of the two real numbers is the distance in km between the towns and the second real number is the estimated maximum average speed with which rescue operations can be executed over the road link.

The input terminates with an input '0' for c. The input is illustrated in sample input.

Output

For each test case in the input first print in one line, the test case number c and the total number of routes in the set F. Print each route in F in two lines. The first line gives the priority number, the operation time and the covering distance while the second line gives the route as a sequence of integers.

Print a blank line between outputs of two successive test cases. The output is illustrated in sample output.

Sample Input

```
1
3
4
1 3 20 20
1 2 10 10
1 4 40 20
1 5 20 0
2 3 30 15
2 4 30 30
2 5 10 5
3 5 40 20
4 5 20 20
2
4 5
3
1 3 20 20
1 2 10 10
1 4 40 20
1 5 20 0
2 3 30 15
2 4 30 30
2 5 10 5
3 5 40 20
4 5 20 20
0
```

Sample Output

```
1 3
1 3.0 60.0
3 1 4
2 3.0 60.0
3 2 4
3 3.0 60.0
3 5 4

2 3
1 2.0 40.0
5 3
2 3.0 60.0
4 2 1 3
3 4.0 40.0
```

```
5 2 3
```