Task 1

Economic times these days are tough, even in Byteland. To reduce the operating costs, the government of Byteland has decided to optimize the road lighting. Till now every road was illuminated all night long, which costs 1 Bytelandian Dollar per meter and day. To save money, they decided to no longer illuminate every road, but to switch off the road lighting of some streets. To make sure that the inhabitants of Byteland still feel safe, they want to optimize the lighting in such a way, that after darkening some streets at night, there will still be at least one illuminated path from every junction in Byteland to every other junction.

What is the maximum daily amount of money the government of Byteland can save, without making their inhabitants feel unsafe?

Input Specification

The input file contains several test cases. Each test case starts with two numbers m and n, the number of junctions in Byteland and the number of roads in Byteland, respectively. Input is terminated by $m=n=0$. Otherwise, $1 \le m \le 200000$ and $m-1 \le n \le 200000$. Then follow n integer triples x, y, z specifying that there will be a bidirectional road between x and y with length z meters ($0 \le x, y < m$ and $x \ne y$). The graph specified by each test case is connected. The total length of all roads in each test case is less than 2^{31}.

Output Specification

For each test case print one line containing the maximum daily amount the government can save.

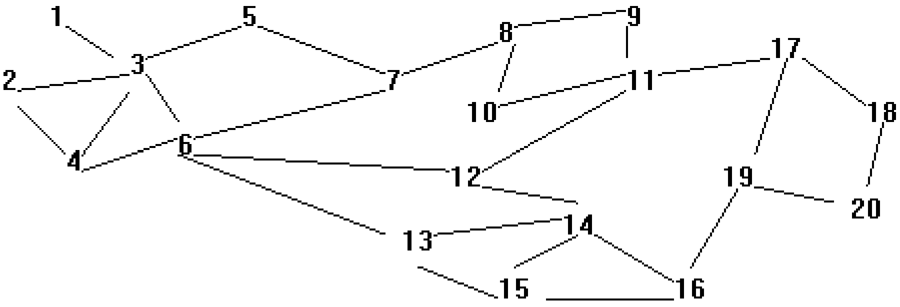
Sample Input	Sample Output

7 11
0 1 7
0 3 5
1 2 8
1 3 9
1 4 7
2 4 5
3 4 15
3 5 6
4 5 8
4 6 9
5 6 11
 	
3 2
0 1 2000000
1 2 1
 	
0 0
	51
	
	0

Task 2

Risk is a board game in which several opposing players attempt to conquer the world. The gameboard consists of a world map broken up into hypothetical countries. During a player's turn, armies stationed in one country are only allowed to attack only countries with which they share a common border. Upon conquest of that country, the armies may move into the newly conquered country. During the course of play, a player often engages in a sequence of conquests with the goal of transferring a large mass of armies from some starting country to a destination country. Typically, one chooses the intervening countries so as to minimize the total number of countries that need to be conquered. Given a description of the gameboard with 20 countries each with between 1 and 19 connections to other countries, your task is to write a function that takes a starting country and a destination country and computes the minimum number of countries that must be conquered to reach the destination. You do not need to output the sequence of countries, just the number of countries to be conquered including the destination. For example, if starting and destination countries are neighbors, then your program should return one.

The following connection diagram illustrates the first sample input.



Input

Input to your program will consist of a series of country configuration test sets. Each test set will consist of a board description on lines 1 through 19. The representation avoids listing every national boundary twice by only listing the fact that country *I* borders country *J* when *I* < *J*. Thus, the *I*th line, where *I* is less than 20, contains an integer *X* indicating how many ``higher-numbered" countries share borders with country *I*, then *X* distinct integers *J* greater than *I* and not exceeding 20, each describing a boundary between countries *I* and *J*. Line 20 of the test set contains a single integer ($1 \leq N \leq 100$) indicating the number of country pairs that follow. The next *N* lines each contain exactly two integers ($1 \leq A, B \leq 20; A \neq B$) indicating the starting and ending countries for a possible conquest.

There can be multiple test sets in the input file; your program should continue reading and processing until reaching the end of file. There will be at least one path between any two given countries in every country configuration.

Output

For each input set, your program should print the following message ``Test Set #*T*" where *T* is the number of the test set starting with 1 (left-justified starting in column 11).

The next N_T lines each will contain the result for the corresponding test in the test set - that is, the minimum number of countries to conquer. The test result line should contain the start country code *A* right-justified in columns 1 and 2; the string `` to " in columns 3 to 6; the destination country code *B* right-justified in columns 7 and 8; the string ``: " in columns 9 and 10; and a single integer indicating the minimum number of moves required to traverse from country *A* to country *B* in the test set left-justified starting in column 11. Following all result lines of each input set, your program should print a single blank line.

Input

```
1 3
2 3 4
3 4 5 6
1 6
1 7
2 12 13
1 8
2 9 10
1 11
1 11
2 12 17
1 14
2 14 15
2 15 16
1 16
1 19
2 18 19
1 20
1 20
5
1 20
2 9
19 5
18 19
16 20


4 2 3 5 6
1 4
3 4 10 5
5 10 11 12 19 18
2 6 7
2 7 8
2 9 10
1 9
1 10
2 11 14
3 12 13 14
3 18 17 13
4 14 15 16 17
0
0
0
2 18 20
1 19
1 20
6
1 20
8 20
15 16
11 4
7 13
2 16
```

Output

```
Test Set #1
 1 to 20: 7
 2 to  9: 5
19 to  5: 6
18 to 19: 2
16 to 20: 2

Test Set #2
 1 to 20: 4
 8 to 20: 5
15 to 16: 2
11 to  4: 1
 7 to 13: 3
 2 to 16: 4
```