UW-Madison ACM ICPC Individual Contest

October 5, 2014

Setup

Before the contest begins, log in to your workstation and set up and launch the PC2 contest software using the following instructions. You will use this program to submit problem solutions, receive the judges answers, and communicate clarification requests.

1. Download the custom PC2 package into a directory of your choosing from www.cs.wisc.edu/~dieter/ICPC/13-14/pc2.tar.gz
2. In a terminal window, cd to the directory where you downloaded the package and type tar -xzvf pc2.tar.gz
3. Type cd pc2 followed by the command bin/pc2team – this brings up your PC2 terminal that will be your interface to the judges during the contest.
4. Log in using the login ID and password given to you by the judges when you arrive. They will be of the form teamX where X is an integer, and the password will be your UVa online judge username. If you have not yet sent your username to the judges, you will not be given a login and password by default – please see the judges to get one.

The Contest

Begin the contest by solving the problem on the next page **"count"**. This is a warmup problem designed to get you used to submitting problems via PC2. Code your solution to the problem and submit it as follows:

1. Click on the submit run tab in your PC2 window.
2. In the dropdown menu labeled "Problem", choose **"count"**. Choose the programming language you used from the "language" dropdown menu. Then select your source code file by clicking the "select" button in the main file section.

3. Submit your code by clicking "Submit" (note: clicking "Test" doesn't really do anything unless you've created your own test files, so don't expect it to automatically test your program against the sample input).
4. Wait – you will receive a judgment from the judge shortly by way of a pop-up window. If your answer comes back something other than "accepted", try again

The remaining problems are known to PC2 as **"Anadromes"**, **"Back on Track"**, **"Twin Head Dragon"**, **"Divide the Galaxy"**, **"Reconstructing the Grid"**, and **"String Painter"** , respective of their order in this packet. You can ignore any requirements stated in the problem for source name file, e.g., "equilateral.cpp": name your source files as you please. All input comes from standard in, all output should be sent to standard out. You may use any online Java or C++ documentation, but not any other resource. You may use the printer at any time. Collaboration with others or searching the web for solutions to these problems is prohibited. Please turn off your cell phones. You may submit problem clarifications via the PC2 program at any time, but please read the problems thoroughly before doing so.

After the contest, please fill out the questionnaire on the next page, then join us in room 1325 for pizza and soda.

Information Form

Name: ____________________

CS Login: __________________________

Student status (e.g. Junior, first year grad student):
____

Year of birth: ________ Year starting college: ________

Which of C/C++/Java do you prefer? ___ Please indicate your proficiency in each language:

What classes have you taken (or are you taking) which are relevant to the ICPC?

What do you feel are your strengths with respect to the ICPC?

Are you able to attend the world finals in Marrakech, Morocco, May 16-21, 2015 (and obtain a Visa and/or passport as necessary)?

Will you be in residence at UW-Madison in the Spring 2015 semester?

How many ICPC regionals have you participated in? ___ How many ICPC world finals? ___

If your team progresses to the world finals, how many hours per week could you commit to practicing? ________

Is there anything else we should know about you?

Warmup Problem: Count

Can you count from one up to any number N? Write a program to prove it!

Input

The input begins with a single number that describes the number of test cases. Each test case follows on its own line, and consists of a single positive integer $N \leq 1,000,000$ that describes how high you should count for that test case.

Output

The output for each test case should be on its own line, and consist of the numbers 1 through N (inclusive), each separated by a space.

Sample Input

```
3
3
5
10
```

Sample Output

```
1 2 3
1 2 3
1 2 3 4 5
1 2 3 4 5 6 7 8 9 10
```

The following pages contain 6 lettered problems. Please let me know if you do not see all 6.

A: Anadromes

A palindrome is a word that is spelt the same backwards or forwards. Examples are “level” and “madam”. An anagram is a word made from another word just by rearranging the characters, like “made” to “dame”. For this problem anagrams and palindromes may not be valid English words. For example “daamm” is an anagram of “madam” and “amdma” is also palindrome although it is not an English word. “daamm” is also called a anadrome because it is the anagram of a palindrome. Any palindrome is also an anadrome but the vice versa is not always true. Some other examples of anadromes are “aabbcc”, “aaaaa” etc. Given a string you will have to print the string that needs to be appended with it to make it an anadrome.

Input

The input file contains at most 6000 test cases. The description of each test case is given below. Each case consists of a single string S of length $L(1 \leq L \leq 500)$. This string contains only lowercase English letters (‘a’ to ‘z’). Input is terminated by a line containing a single hash (‘#’) character. This line need not be processed.

Output

For each line of input produce one line of output. The line contains the string that needs to be appended to the given string to make it an anadrome (Can be an empty string as well). If there is more than one solution, output the shortest one; if there is still a tie output the lexicographically smallest one.

Sample Input

```
ddc
aaab
#
```

Sample Output

```
a
```

B: Back on Track

The road system of a country connects all N cities so that it is possible to travel between any pair of cities using existing roads. Each road connects two different cities, is two-way and one has exactly one toll booth (a toll is paid for both directions of traffic). Roads intersect only in a city and no pair of cities is interconnected by two or more roads.

Dias Tranport offers a one-day parcel delivery service between cities. Each parcel must be transported from a city A to another city B. The management of Dias Transport defines, for each parcel, a service route, consisting of C cities and $C - 1$ roads: the first city on the service route is the origin of the parcel, the final city is the destination of the parcel. The service route never passes twice through the same city, and the vehicle chosen to deliver a parcel can only travel by the service route defined.

One day, however, a vehicle broke down and was taken for repairs in a city that was not among the cities in its service route. The management of Dias Transport wants to know which is the lowest total cost, in terms of tolls, for delivering the parcel (that is, to take the vehicle from the city it was repaired to the destination city), but with an additional constraint: if at some point the vehicle reaches one of the cities that make up its service route, it should go back to following its service route.

Input

The input contains several test cases. The first line of a test case contais four integers N, M, C, and K ($4 \leq N \leq 250, 3 \leq M \leq N \cdot (N-1)/2, 2 \leq C \leq N-1, C \leq K \leq N-1$), representing, respectively, the number of cities, the number of roads, the number of cities in the service route and the city where the vehicle was taken for repair. The cities are identified by integers from 0 to $N-1$. The service route is $0, 1, \ldots, C-1$, that is, the origin is 0, from 0 goes to 1, from 1 to 2 and so on, until the destination $C-1$. The next M lines describe the road system. Each of those lines describes one road and contains three integers U, V and P ($0 \leq U, V \leq N-1, U \neq V, 0 \leq P \leq 250$), indicating that there exists a road connecting cities U and V with a toll of cost P.

The last test case is followed by a line containing four zeros separated by blank spaces.

Output

For each test case, your program should print a single line, containing a single integer, the minimum total toll cost for the vehicle to reach the destination city.

Sample Input

```
4 6 3 3
0 1 10
```

```
1 2 10
0 2 1
3 0 1
3 1 10
3 2 10
6 7 2 5
5 2 1
2 1 10
1 0 1
3 0 2
3 4 2
3 5 3
5 4 2
5 5 2 4
0 1 1
1 2 2
2 3 3
3 4 4
4 0 5
0 0 0 0
```

Sample Output

```
10
6
6
```

C: The Twin Head Dragon

The Scourge are marching South-West with the biggest army ever seen, and they're marching fast. All the Sentinel towers are in ruins. There's chaos all over their base. Whatever they have to do, they have to do it by tonight or they will be terminated from the face of the earth tomorrow. A secret meeting is being conducted by Zeus, the lord of Olympus and the father of gods. "The end is near." dreads Sven. "Careful child. Sentinel will not be doomed before my eyes." says Zeus. "Send Riki to explore the enemy camps. Then we can come up with a plan."

Riki comes with the information that there are N enemy camps and M bi-directional roads, each road connecting two camps. The lengths of the roads are different. Riki also found that there exists a path between any pair of camps. "That's enough information!" says Zeus with excitement, "We have to burn down all the roads, so the enemies will be isolated from each other. Then we will strike. Summon Jakiro, he'll know what to do." Jakiro, the Twin Head Dragon is summoned. He will use his ultimate spell, Macropyre to burn all the roads down. To do this he will follow these steps:

1. Select 2 different camps such that the shortest path between them doesn't include any burned road.
2. Prepare the spell with required mana. The mana cost for this spell is equal to the sum of the lengths of roads in the path.
3. Burn all roads in the selected path.

He will keep burning this way until all shortest paths include roads are burning. It is important that he uses minimum total mana for this task, as he needs mana for the battle afterwards. Now write a program to calculate the least mana required by Jakiro to burn all the roads down. Remember, you don't need to minimize the number times the spell is used.

Input

The input will contain multiple test cases and number of test cases ≤ 50. Each case starts with an integer $N (2 \leq N \leq 15)$ denoting the number of enemy camps and an integer $M (N-1 \leq M \leq N \cdot (N-1)/2)$ denoting the number of roads connecting camps. The camps are numbered from 0 to $N-1$. Each of the next M lines contain three integers $A, B, C : (0 \leq A, B < N, A \neq B, 1 \leq C \leq 10000)$ denoting that camp A and B are connected by a road whose length is C units. You may assume that all pairs of A and B are unique.

The input terminates with values of 0 for N and M.

Output

For each case, print on a line the least total mana required by Jakiro.

Sample Input

```
4 5
0 2 1
1 2 2
2 3 3
3 1 4
0 1 5
6 5
0 1 10000
0 2 10000
0 3 1
0 4 1
0 5 10000
0 0
```

Sample Output

```
10
30002
```

D: Divide the Galaxy

The Galactic Confederation is planning an administrative reform, to better manage its resources. For that, the Confederation divided the whole space into regions. To define the regions, initially a set of planes was specified, and the regions were defined by the cuts these planes made in the space. Notice that some regions are unlimited, but there may be limited regions. The set of planes was chosen so that no plane intercepts the orbit of a planet, and therefore each planet moves within only one region during its orbit (that is, a planet inside a region will never cross a plane to another region). Your task is to determine, given the equations of the planes and the positions of the planets, how many planets exist within the region with the largest number of planets (in other words, what is the maximum number of planets inside any region).

Input

The input contains several test cases. The first line of a test case contains two integers $M (1 \leq M \leq 500)$ and $N (1 \leq N \leq 10000)$, indicating respectively the number of planes and the number of planets. Each of the M following lines contains four integers A, B, C and $D (-10000 \leq A, B, C, D \leq 10000)$, the coefficients and the free term of the equation $Ax + By + Cz = D$ which defines each plane. Each of the following N lines contains three integers X, Y and $Z (-10000 \leq X, Y, Z \leq 10000)$, representing the position (X, Y, Z) of a planet.

Output

For each test case in the input your program must produce a single line containing a single integer, the number of planets in the region which contains the largest number of planets.

Sample Input

```
2 5
1 0 0 1
2 0 0 8
0 1 0
2 2 2
3 3 3
5 5 5
2 18 4
4 8
0 0 1 1
1 0 1 2
-1 1 1 3
-1 -1 1 3
0 0 5
```

```
0 0 4
0 0 -2
1 0 5
40 19 104
13 26 84
89 -45 18
3 1 0
```

Sample Output

```
3
5
```

E: Reconstructing the Grid

Fox Shial loves to collect grids of numbers. One of his favorite grids had been stolen recently. It had R rows and C columns and the grid had every integers in the range 1 to $R * C$ exactly once in some arbitrary order. For each integer n in the range 1 to $R * C$ (inclusive), Fox Shial remembers the numbers that were adjacent to n in the stolen grid. A cell $(x; y)$ is adjacent to at most four other cells $(x - 1; y), (x + 1; y), (x; y - 1), (x; y + 1)$. Your task is to reconstruct the grid for Shial. If there are multiple possible grids, find the one that is lexicographically smallest. A grid G_1 is lexicographically smaller than some other grid G_2 if the following condition holds true: If we traverse both of the grids in the row major order and if $(x; y)$ is the first cell where the $G_1[x][y] \neq G_2[x][y]$, then $G_1[x][y] < G_2[x][y]$. (Here, $(x; y)$ denotes the cell at row x and column y). Note: Any cell $(x_1; y_1)$ comes before $(x_2; y_2)$ in a row major order, if and only if either $(x_1 < x_2)$ or $(x_1 == x_2$ and $y_1 < y_2)$ holds true.

Input

The first line contains an integer T denoting the number of test cases. Each test case begins with a line containing 2 integers R and C where R is the number of rows and C is the number of columns in the stolen grid. Each of the next $R * C$ lines contains a list of numbers. The i^{th} line starts with an integer k_i and then k_i distinct space-separated integers follow. All these integers will be in the range 1 to $R * C$ (inclusive). Here k_i is the number of integers adjacent to the number i in the stolen grid. The numbers following k_i are all of those adjacent integers in an arbitrary order. It is guaranteed that, if some integer u is adjacent to some other integer v , then v is also adjacent to u. No integer is adjacent to itself.

Constraints:
$1 \leq T \leq 40$
$1 \leq R, C \leq 100$
$0 \leq ki \leq 4$

Output

For the output of each input case, print the serial of the input on a single line and then print the grid in the following format. Each row should be printed on a different line. Every number of a row should be printed with exactly 1 space between the numbers. There should be no space at the end of a row. (See the sample input output). If the given input is invalid (i.e. there is no grid that satises the given adjacency information) print 'NO SUCH GRID' (without quote).

Sample Input

```
2
```

```
2 2
2 3 4
2 3 4
2 1 2
2 1 2
1 3
2 2 3
2 1 3
2 1 2
```

Sample Output

```
Case 1:
1 3
4 2
Case 2:
NO SUCH GRID
```

F: String Painter

There are two strings A and B with equal length. Both strings are made up of lower case letters. Now you have a powerful string painter. With the help of the painter, you can change a segment of characters of a string to any other character you want. That is, after using the painter, the segment is made up of only one kind of character. Now your task is to change A to B using string painter. What's the minimum number of operations?

Input

Input contains multiple cases. Each case consists of two lines:

- The first line contains string A.
- The second line contains string B.

The length of both strings will not be greater than 100.

Output

A single line contains one integer representing the answer.

Sample Input

```
zzzzzfzzzzz
abcdefedcba
abababababab
cdcdcdcdcdcd
```

Sample Output

```
6
7
```