Premio Fase II México & Centroamérica
The ACM-ICPC in ESCOM-IPN 2015

October 10th, 2015

Problemset

Authors:

Edgar Augusto Santiago Nieves

Ethan Adrian Jiménez Vargas

Yonny Mondelo Hernández

Document composed of 14 pages including this cover.

Problem A. Alcoholic Pilots

Time Limit:	1 second
Stack Limit:	10 MB
Memory Limit:	32 MB

In one of his many trips, Mr. Ed boarded an airplane where the captain talked like… well, like he was completely drunk. "I would like to greet a very special person here, which has been part of our lives for so much time. His wife says from the control tower that she loves you" – the captain said. Of course, Mr. Ed was really scared, how can alcoholic pilots flight with so many people on their hands? But that was not the worst part, our friend noticed that these drunken pilots like to race between them!

By getting close to the captain's cabin, Mr. Ed could hear another pilot (drunk, as expected) discussing with the captain by radio. Both of them shared information about how fast they were travelling and how far they were to the nearest airport. "If I arrive earlier to the airport, you will owe me a beer" – the captain bragged, then the airplane started to move abruptly.

Of course Mr. Ed survived, if not, he could not tell us this story. But weirdly, you are wondering who won the race between the pilots and their average arrival time, so you asked the velocity and distance to the airport of both planes. Assume that the planes maintained their velocity even when landing.

Input

The input will contain several test cases. The only line of each test case contains 4 space-separated integers v_1, d_1, v_2 and d_2 ($1 \leq v_1, d_1, v_2, d_2 \leq 10^9$): the velocity and distance to the airport of the plane Mr. Ed and the captain were and the velocity and distance to the airport of the plane the captain was competing with. Velocities are expressed in miles per hour and distances in miles.

The last test case is followed by a single line containing 4 zeroes.

Output

Print 2 lines for each test case. In the first one, you should print "`You owe me a beer!`" if the captain won the race or "`No beer for the captain.`" if the other airplane won the race.

You can safely assume there will be no draws in any test case.

In the second line, print "`Avg. arrival time:`" followed by the average arrival time (in hours) of both airplanes expressed as a simplified fraction of the form x/y, being x and y integers. If the fraction has an integer result, print the result of the division. See format below for more details.

Example

Input	Output

2 4 1 3
1 3 2 4
4 7 4 9
0 0 0 0
	Case #1: You owe me a beer!
	Avg. arrival time: 5/2
	Case #2: No beer for the captain.
	Avg. arrival time: 5/2
	Case #3: You owe me a beer!
	Avg. arrival time: 2

Problem B. Bargaining

Time Limit: 3 seconds
Stack Limit: 10 MB
Memory Limit: 32 MB

When he was in the old medina of Marrakech, Mr. Ed and his pals visited a famous Berber market. There were several vendors offering all kind of species, cloths, babouches, lamps and tea pots. As you may imagine, Mr. Ed wanted to buy many souvenirs for all his family. He approached to one of the vendors and asked the price for a pair of babouches, then, the vendor asked "How much are you willing to pay?" Mr. Ed understood that he needed to use his business abilities to set a fair price. "I'll not pay more than 30 euros for that" – said Mr. Ed, "Let me make you a deal, you give me 35 euros, I give you back 10 dirhams and the babouches are yours" – replied the vendor.

Mr. Ed accepted the previous deal and later noticed that he had been cheated! Furious, Mr. Ed decided that he would not be cheated again, so he started to bargain with the vendors to gather information about the prices for a pair of babouches. After several minutes, he managed to bargain with n vendors.

The i-th vendor he bargained with gave euros a value of e_i and dirhams a value of d_i. After realizing that, he came up with an inequality of the form: $e_i\,EUR \pm d_i\,MAD > c_i$, meaning that the total value of euros and dirhams Mr. Ed pays must be more than c_i if he expects the vendor to accept; or $e_i\,EUR \pm d_i\,MAD < c_i$, meaning that he only accepts if the total value of euros and dirhams he pays is less than c_i.

After gathering such information Mr. Ed was ready to buy again. He still had E euros and D dirhams in his pocket, but now you're wondering: which was the effective area of prices he could bargain with that money? The effective area of prices is the area of every possible way Mr. Ed could buy the babouches satisfying the n inequalities, assuming any positive real number of euros and dirhams.

Input

The input will contain several test cases. The first line of each test case contains 3 integers E, D and n, representing the euros and dirhams Mr. Ed had and the number of inequalities gathered ($1 \le E, D \le 1{,}000$ and $0 \le n \le 1{,}000$). The next n lines contains an inequality as shown in the example, the values for e_i, d_i and c_i will be integers that satisfy $0 \le e_i, d_i \le 10{,}000$ and $0 \le |c_i| \le 10{,}000$.

The last test case is followed by a single line containing 3 zeroes.

Output

For each test case print a real number with exactly 2 digits after the decimal point, representing the effective area of prices that Mr. Ed could bargain with the vendor (see format below).

Example

Input

```
2 2 2
1EUR + 0MAD > 1
1EUR + 0MAD < 1
2 2 2
1EUR + 1MAD < 2
1EUR - 1MAD > 1
0 0 0
```

Output

```
Case #1: 0.00
Case #2: 0.25
```

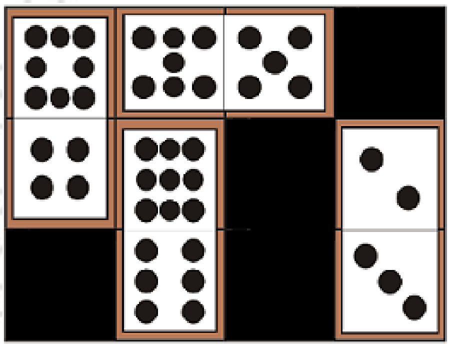
Please note that in this problem there is no relation between Euros (EUR) and Moroccan Dirhams (MAD).

Problem C. Cuban Challenge

Time Limit:	3 seconds
Stack Limit:	10 MB
Memory Limit:	32 MB

Mr. Ed is visiting his pals at Cuba; in particular he wants to have a nice chat with Yonny, who is a very talented domino player. After taking a few rounds of rum, everybody got ready to play domino. As Mr. Ed and Yonny are close friends, they make an invincible team at domino. They won every single game against their friends, so they started to get bored of playing. “Do you feel you play domino as good as a true Cuban?” – asked Yonny. “I play even better” – presumed Mr. Ed.

Now Yonny has challenged Mr. Ed to complete the following task: “I have a wood board divided in n rows of m columns such that there are $n \times m$ squares of equal size on it. The challenge is simple: use as many dominoes as you want to cover each single square in the board without overlapping, but not so fast, that will be pretty easy, right? There are some squares colored in black, you cannot put a domino on this squares. Here, let me show you an example.”



“Just because you are my friend, I will let you cut some dominoes in half if you have trouble completing the challenge, but for every domino you cut, you shall pay a small fee to buy more dominoes for me.”

As you may expect, now Mr. Ed is all fired up trying to beat Yonny’s challenge, so he is trying to answer: which is the minimum number of dominoes he needs to cut in order to fill every single non-black square in the $n \times m$ wood board?

Input

The input will contain several test cases. The first line of each test case contains 2 integers n and m ($1 \le n \le 20$ and $1 \le m \le 1{,}000$), representing the number of rows and columns in the board. Each of the next n lines contains m characters describing each square in the wood board: `'.'` means there is an empty square and `'#'` means there is a black square in the wood board.

The last test case is followed by a single line containing 2 zeroes.

Output

For each test case, print the number of required cuts to complete the Cuban challenge (see format below).

Example

Input	Output
3 4 ...# ..#. #.#. 3 4 ...# ..#. ##.. 0 0	Case #1: 0 Case #2: 1

Problem D. Drinking Game

Time Limit:	3 seconds
Stack Limit:	10 MB
Memory Limit:	32 MB

In the way back home from the World Finals, Mr. Ed and his pals are visiting Madrid, but Mr. Ed is actually very tired and decided to rest a few days, so this problem is not about him.

Ethan and the gang are going to drink some shots in the Gran Via; there, they have a good time drinking and talking about monsters and other terrifying adventures. After several minutes, Ethan came up with a fun drinking game he heard about at the World Finals called "Coprime Shots", he explains:

"Everybody has n piles of empty shot glasses, numbered from 1 to n, the i-th pile has g_i glasses piled up. The objective of the game is to add or remove an arbitrary number of glasses to some of your piles in a way such that, for any pair of distinct piles (i, j) in the set G (result of adding and removing glasses to the original n piles), it holds that $\gcd(G_i, G_j) = 1$. Sounds cool, huh? Anyway, you better play smart, because for every single shot glass you add or remove, you have to fill it up and drink!"

Being a decent programmer, you have no plans on getting wasted tonight, so you decided to minimize the number of shots you have to take in order to win the game.

Input

The input will contain several test cases. The first line of each test contains an integer n: the number of piles in the game ($1 \leq n \leq 100$). The next line contains n integers: the i-th integer represents g_i, the number of shot glasses in the i-th pile ($1 \leq g_i \leq 20$).

The last test case is followed by a single line containing 1 zero.

Output

For each test case, print n positive integers separated by a single space, describing a winning configuration of piles that requires the minimum number of additions and removals to accomplish. Piles should be printed in the same order as in the input. See details in the format below.

If there are multiples solutions to a test, any one of them will be accepted.

Example

Input	Output

`3`
`1 2 3`
`5`
`2 4 6 9 10`
`0`
	`Case #1: 1 2 3`
	`Case #2: 1 4 7 9 11`

Problem E. Exquisite Strings

Time Limit:	3 seconds
Stack Limit:	10 MB
Memory Limit:	32 MB

Mr. Ed is a very sophisticated man and likes art very much, that is why he and his pals are visiting the amazing museums of Paris. His favourite museum is the Musée d'Chaîne; it has a huge art collection, but the masterpiece of the exhibition is a world famous string of n characters. Mr. Ed has spent hours and hours looking for exquisite pairs of substrings inside the masterpiece.

A substring of a string $S = s_1 s_2 \ldots s_n$, represented as $T_{i,j}$ for a pair of indexes $i \leq j$, is described as the concatenation $s_i s_{i+1} \ldots s_{j-1} s_j$ of characters from string S. Two substrings of S are considered distinct if their indexes i and j are not the same.

The group does not want to observe a single string all day long. In order to leave the museum as soon as possible, you want to help Mr. Ed counting every pair of distinct substrings of the exhibition string that are exquisite. If you don't have as much artistic taste as Mr. Ed, a pair of strings is considered exquisite if they share a common prefix of at least k characters.

If you don't have idea what a prefix is *sigh*, we define it as a substring with starting index i equal to 1.

Input

The first line of input contains a positive integer T representing the number of test cases.

The following T lines contain a non-empty string of $1 \leq n \leq 10^5$ lowercase letters of the English alphabet representing the museum exhibition string, followed by an integer number $1 \leq k \leq n$; the length of the minimum prefix required in an exquisite pair of strings.

Output

For each test case in the input, print a single line with an integer representing the number of exquisite substring pairs, modulo 1,000,000,007 ($10^9 + 7$). See format below for details.

Example

Input	Output
5 aaaa 3 ababab 4 cdabcdab 5 qwertyuiop 2 abcabcabcabcx 5	Case #1: 3 Case #2: 7 Case #3: 10 Case #4: 120 Case #5: 313

Problem F. File Transmission

Time Limit:	3 seconds
Stack Limit:	10 MB
Memory Limit:	32 MB

As you may know, Mr. Ed is a very important international businessman. He owns many data centers all around the world, where he keep critical files for his international operations. More specifically, Mr. Ed owns n data centers, numbered from 1 to n. Among these data centers, there are network connections that allow both endpoints of the connection to transfer files with a bandwidth of 50 GB. To ensure complete access to his files, the data centers are connected in such a way that, for any pair (a, b) of distinct data centers, there is at least one way of reaching b from a.

This global network is so powerful that transferring a file along any connection takes no time. Anyway, Mr. Ed knows that the transmission time is no problem in his network, the real problem occurs when he tries to move a file that exceeds the bandwidth of a connection. Explicitly, Mr. Ed cannot transfer instantly a file of 100 GB along a single connection in the network; he will need to transfer one half of the file in one instant and the other half in another instant of time.

Fortunately, data centers can segment a file in multiple parts in order to distribute the transfer load among multiple paths. That way, one file of 100 GB may be transferred instantly. Notice that there might be no way of segmenting a file to achieve this all the time.

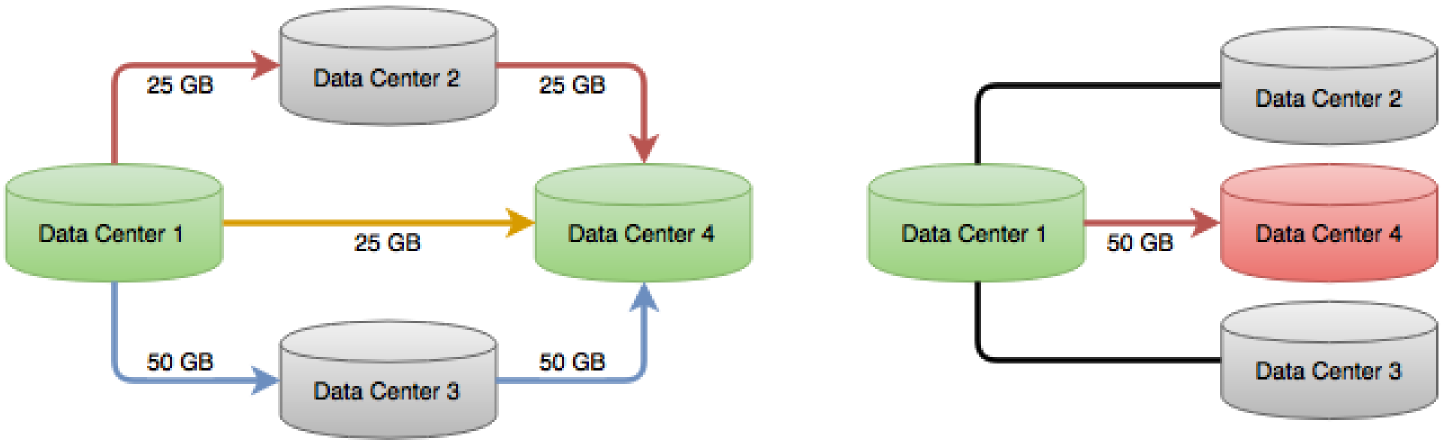


Figure 1. One way of segmenting a file to transfer it from data center 1 to 4 instantly (left).
There is no way to instantly transfer a file of 100 GB from 1 to 4 (right).

To ensure that always is possible to instantly transfer a file of 100 GB between data centers, Mr. Ed can increase (by 1 GB) the bandwidth of any connection by paying 1 dollar.

Mr. Ed do not like to pay for more than he requires, so he asked you to write a software that, given the description of his n data centers and the network connections between them, answers several queries of the type: which is the minimum number of dollars I need to pay in order to assure that I can instantly transfer a file of 100 GB from data center u to data center v?

Input

The input will contain several test cases. The first line of each test case contains 2 integers n and m ($1 \leq n \leq 10{,}000$ and $1 \leq m \leq 20{,}000$): the number of data centers and the number of connections.

The next m lines contains 2 integers a and b each ($1 \leq a, b \leq n$), representing that there is a network connection between data center a and data center b. The will be no connections from any data center to itself and there will be at most one connection between any two data centers.

The next line contains 1 integer q ($1 \leq q \leq 20{,}000$): the number of queries you need to answer. Following this, there will be q lines with 2 integers u and v each ($1 \leq u, v \leq n$), representing the source data center and target data center for the q queries. Test cases end with a blank line.

The last test case is followed by a single line containing 2 zeroes.

Output

For each test case, print the case number followed by q lines with the answer to every query on the test case. Print a single blank line between cases (see format below).

Example

Input

```
4 5
1 2
2 4
1 3
3 4
1 4
2
1 4
3 2

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

0 0
```

Output

```
Case #1:
0
0

Case #2:
50
100
```

The example test cases show the networks illustrated in figure 1. Test case number 1 is the network shown in the left and the second shows the example network in the right.

Problem G. Greedy Artisan

Time Limit:	1 second
Stack Limit:	10 MB
Memory Limit:	32 MB

On their way to the next World Finals, Mr. Ed and his pals are visiting the beautiful city of Moscow. One of their favorite tourism activities is buying souvenirs to bring back home, so they are looking for matryoshkas in a big artisan market close to the Red Square.

In the market, there is a very greedy and clever artisan that sells custom sets of matryoshkas. This artisan has n different matryoshkas in stock, each one having a unique identifier i $(1 \le i \le n)$, a size s_i and a base price p_i. As the artisan is really clever, he offers a special deal to his clients:

Assume someone wants to buy the custom set $T = \{i_1, i_2, \dots, i_m\}$ of m matryoshkas. Let us call i_{max} to the identifier of the matryoshka with the maximum size and, in case there are multiple matryoshkas with maximum size, the maximum price in T, then the price one has to pay to buy T is

$$price(T) = \sum_{j=1}^{m} \frac{s_{i_j}}{s_{i_{max}}} \times p_{i_{max}}$$

Mr. Ed wants to exploit the artisan's deal buying exactly k matryoshkas, regardless which are the sizes of each matryoshka. Please determine the minimum number of money he needs to expend.

Input

The input will contain several test cases. The first line of each test case contains 2 space-separated integers n and k, representing the number of matryoshkas the artisan has in stock and the number of matryoshkas Mr. Ed wants to buy $(1 \le n \le 100{,}000$ and $1 \le k \le n)$.

There will follow n lines. The i-th line contains 2 integers s_i and p_i, representing the size and the base price of the i-th matryoshka $(1 \le s_i, p_i \le 10^6)$. There may be matryoshkas with the same s_i and p_i.

The last test case is followed by a single line containing 2 zeroes.

Output

For each case, print a single line with a real number with 6 digits after the decimal point representing the minimum price Mr. Ed has to pay to buy k matryoshkas (see format below).

Example

Input	Output

`3 2`
`10 5`
`4 4`
`6 3`
`0 0`
	`Case #1: 5.000000`

Problem H. Heavy Luggage

Time Limit:	9 seconds
Stack Limit:	10 MB
Memory Limit:	32 MB

Travelling around the world is really tiring, especially for Mr. Ed and his pals, who carry heavy luggage.

In order to reduce the fatigue, they planned to share the weight of everyone's luggage between their friends. Let's say that person i was carrying w_i kilograms of luggage and this person has f_i friends in the group, then he distributed equitably that weight such that every friend received exactly w_i/f_i kilograms from him. Nobody distributed luggage they had just received.

At the first day of a trip, they distributed the luggage everyone brought from home; by the second day they distributed the luggage received on day one distribution; by the third day of the trip, received luggage from day two is shared; and so on. They kept doing this while they were travelling.

When Mr. Ed arrived home from his latest trip to the World Finals, he noticed that the group forgot to return everyone's luggage! He remembers that there was n people (numbered from 1 to n) travelling in the group, the trip lasted k days and everyone brought a real non-negative number of kilograms at the beginning of the trip. After calling everyone by phone, Mr. Ed wrote down the list of everybody's friends and how many kilograms of luggage they ended up with, including his.

Mr. Ed is exhausted from the trip, so he asked you to find how many luggage each one initially brought.

Input

The input will contain several test cases. The first line of every test case will contain 3 integers n, m and k: the number of people in the group, the number of friendship relations and the number of days the trip lasted ($2 \leq n \leq 16$, $n \leq m \leq n \times (n-1)$ and $1 \leq k \leq 64$).

Each of the next n lines contains a single real number $0 \leq w_i \leq 1600$ (with up to 200 digits to the right of the decimal point): the kilograms of luggage the i-th person ended up with.

The next m lines contain 2 integers a and b ($1 \leq a, b \leq n$ and $a \neq b$), each line describing a friendship relation such that person a considers person b a friend. Notice that relations may not be mutual. There will not be repeated relations and every person will consider at least one friend.

The last test case is followed by a single line containing 3 zeroes.

Output

For each test case print n numbers; the i-th number represents the kilograms of luggage person i brought initially to the trip, rounded (half up) to the nearest integer value. You can safely assume that there is at least one solution for each test case, but if there are multiple solutions you must print "`Lost luggage!`" See example below for details about output format.

Example

Input	Output

2 2 7
1.00
1.00
1 2
2 1
3 3 2
1.00
2.00
3.00
1 2
2 3
3 1
3 4 1
1.50
2.00
1.50
1 2
2 1
2 3
3 2
0 0 0
	Case #1: 1 1
	Case #2: 3 1 2
	Case #3: Lost luggage!

First test case is pretty straightforward; both people in the group are mutual friends and they alternated their luggage for 7 days, ending up with 1 kg of luggage each.

For the second test: initially person 1 brought 3 kg of luggage, person 2 brought 1 kg and person 3 brought 2 kg. Person 1 considers person 2 a friend, while person 2 considers person 3 a friend and this last one considers person 1 a friend. After one day of the trip, person 1 gives his initial 3 kg to person 2, this one gives 1 kg of luggage to person 3 and similarly he gives 2 kg to person 1. By the second day, person 1 gives the 2 kg he received in the previous day to person 2, this one gives last day 3 kg to person 3 and finally person 3 passed his 1 kg of luggage to person number 1. This is the only way person 1, 2 and 3 could end up with 1, 2 and 3 kilograms respectively.

There are multiple ways third case result could be achieved, one of them being: person 1 brought 2 kg of luggage, person 2 brought 3 kg and person 3 didn't bring any luggage to the trip.

Problem I. Incredulous Ed

Time Limit:	1 second
Stack Limit:	10 MB
Memory Limit:	32 MB

Today Mr. Ed lent his phone to Ethan. He didn't lock his phone with password because he thought Ethan is a reliable guy, but he wrote "I'm gay" on Ed's timeline. Obviously Mr. Ed is very pissed.

Due to the recent betrayal to his trust, Mr. Ed wants to lock his phone with a pattern. You know what a phone unlock pattern is, right? The lock screen of a phone consists in a grid of n rows and m columns of dots (or circles). Mr. Ed can choose any sequence of dots $S = d_1, d_2, \dots, d_k$ as the phone unlock pattern; this means that, in order to unlock the phone, Ed needs to trace a path starting in d_1 and ending in d_k that passes along every dot in S in exactly the same order.

There are two restrictions to the sequence of dots S: first, every dot d_i must appear at most one time in the sequence and; for every two consecutive dots in S, let's call them d_i and d_{i+1}, the straight line from dot d_i to d_{i+1} must not pass by any dot not previously visited in the sequence. For example, in a 1×3 grid, the sequence $S = 1,3,2$ is not possible since the straight line from 1 to 3 passes through 2 (not visited yet).

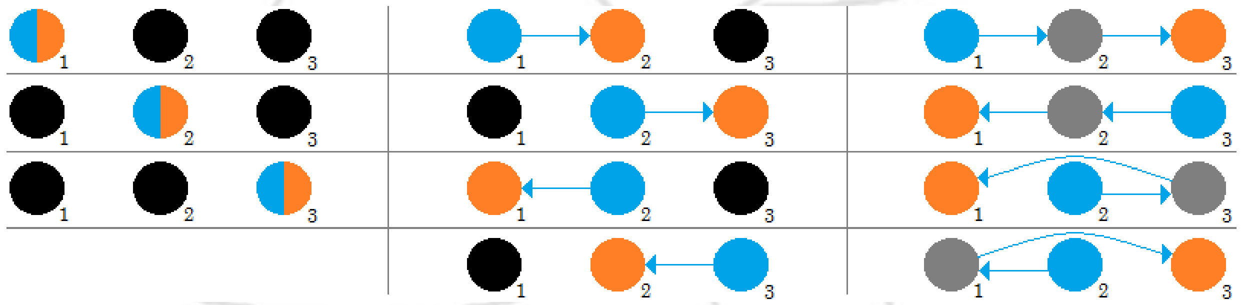


Figure 2. Possible unlock patterns for a 1×3 grid using 1 dot (left), 2 dots (middle) and 3 dots (right).

The above figure illustrates the 11 possible unlock patterns for a lock screen of 1×3 dot grid; blue dots represent the starting dot of the pattern and orange dots represent the respective end dot.

Please help Mr. Ed counting the number of unlock patterns he could use to lock his phone.

Input

The input will contain several test cases. Each test case consists of a single line with two integer numbers $1 \le n \le 3$ and $1 \le m \le 3$: the number of rows and columns in the lock screen grid. The last test case is followed by a single line containing 2 zeroes, which should not be processed.

Output

For each test case, print the number of possible unlock patterns Mr. Ed could use (see format below).

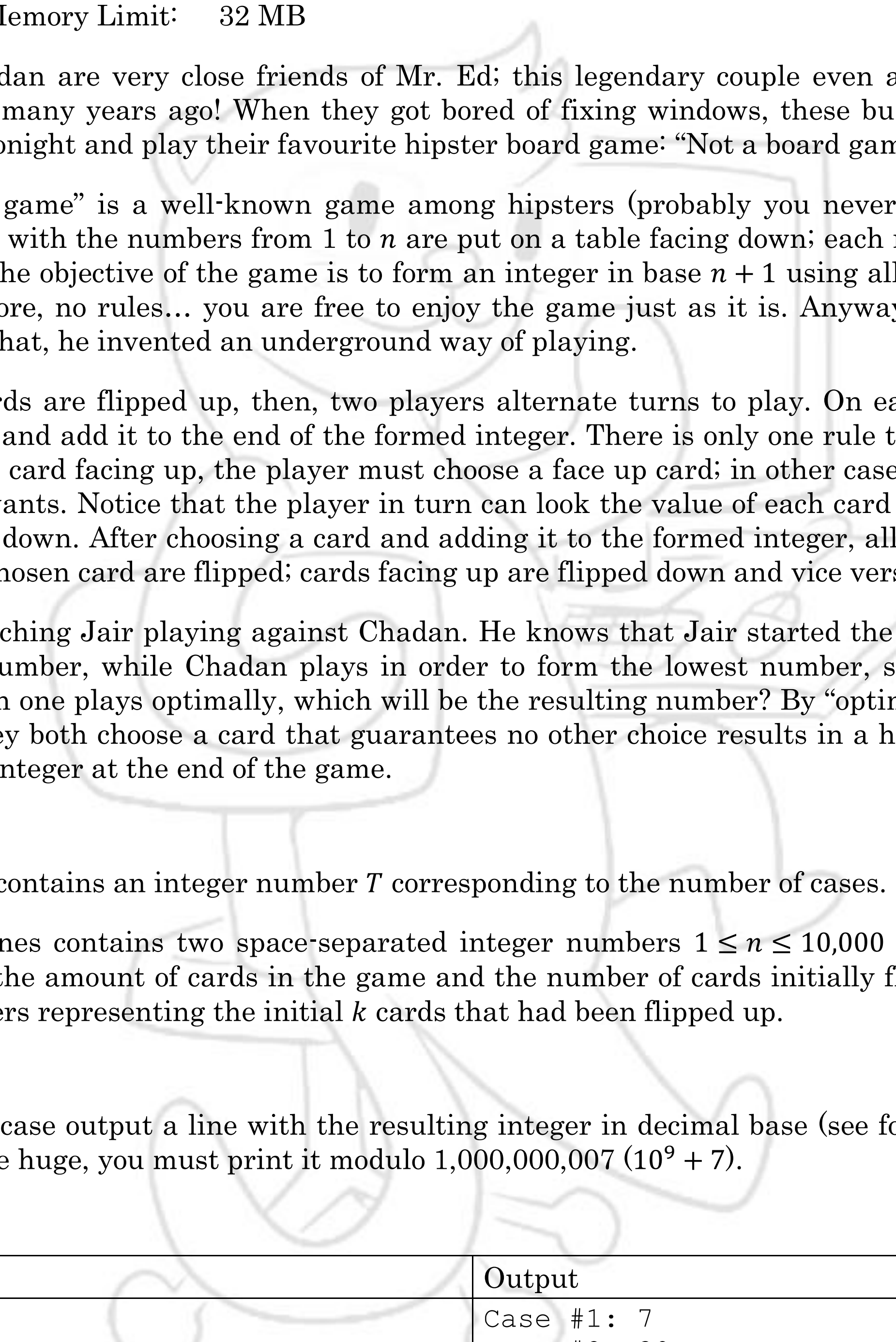
Example

Input	Output

2 1
1 3
0 0
	Case #1: 4
	Case #2: 11

Problem J. Jair vs Chadan

Time Limit:	3 second
Stack Limit:	10 MB
Memory Limit:	32 MB



Jair and Chadan are very close friends of Mr. Ed; this legendary couple even accompanied him to the World Finals many years ago! When they got bored of fixing windows, these buddies invited Mr. Ed to have dinner tonight and play their favourite hipster board game: "Not a board game".

"Not a board game" is a well-known game among hipsters (probably you never hear about it). In this game, n cards with the numbers from 1 to n are put on a table facing down; each number appears exactly in one card. The objective of the game is to form an integer in base $n + 1$ using all the cards… there is no winner, no score, no rules… you are free to enjoy the game just as it is. Anyway, Chadan is even more hipster than that, he invented an underground way of playing.

Initially k cards are flipped up, then, two players alternate turns to play. On each turn, a player must choose a card and add it to the end of the formed integer. There is only one rule to choose a card: if there is at least one card facing up, the player must choose a face up card; in other case, the player may choose any card he wants. Notice that the player in turn can look the value of each card in the table, even if the card is facing down. After choosing a card and adding it to the formed integer, all the cards with a prime factor of the chosen card are flipped; cards facing up are flipped down and vice versa.

Mr. Ed is watching Jair playing against Chadan. He knows that Jair started the game and aims to form the highest number, while Chadan plays in order to form the lowest number, so he is now wondering: assuming each one plays optimally, which will be the resulting number? By "optimally" we mean that, on each turn, they both choose a card that guarantees no other choice results in a higher (for Jair) or lower (for Chadan) integer at the end of the game.

Input

The first line contains an integer number T corresponding to the number of cases.

The next T lines contains two space-separated integer numbers $1 \leq n \leq 10{,}000$ and $1 \leq k \leq \min(100, n)$ representing the amount of cards in the game and the number of cards initially flipped up, followed by k distinct integers representing the initial k cards that had been flipped up.

Output

For each test case output a line with the resulting integer in decimal base (see format below). Since this number can be huge, you must print it modulo 1,000,000,007 ($10^9 + 7$).

Example

Input	Output

`3`
`2 0`
`3 1 2`
`7 5 1 2 4 6 7`
	`Case #1: 7`
	`Case #2: 39`
	`Case #3: 1894165`

Problem K. Keypad Problem

Time Limit:	3 seconds
Stack Limit:	10 MB
Memory Limit:	32 MB

Back in the Moscow artisan market, Mr. Ed bought an antique calculator which he really wanted to show to his pals. When he came back, he noticed that the keypad of the calculator is incredibly strange!

Instead of having the usual keys from 0 to 9, this weird calculator has n different numeric keys and three extra keys with the operators '+' (plus), '–' (minus) and '=' (equals). The i-th key has a number k_i labeled on it, meaning that by pressing this key, the number k_i is showed on the calculator's screen. If there is some other number currently displayed on the screen, it is replaced by k_i.

As you may notice, this is pretty impractical, but Mr. Ed is clever and knows that in order to display a number not labeled in the keypad, he can make some additions and subtractions of the available numbers. For example, suppose Mr. Ed wants to display the number 7 and the available numeric keys of the keypad are 1, 3 and 5. Our friend could press the third key (with label 5), then press the '+' key (to add another number) followed by the second key (adding 3), press the '–' and first key (in order to subtract 1) and finally get the result of the operations by pressing the '=' key, summing up the number 7.

Mr. Ed wants to display the number r in the calculator's screen, but he is not even sure he can achieve that. Please write a program that tells Mr. Ed if he can display number r and, in case it is possible, shows how to add up r by pressing several keys of the keypad.

Input

The input will contain several test cases (up to 100). The first line of each test contains 2 integers n and r: the number of numeric keys in the keypad and the integer Mr. Ed wants to display ($1 \leq n \leq 50$ and $1 \leq |r| \leq 10^{18}$). The next line contains n integers; the i-th integer represents the number k_i labeled in the i-th numeric key ($1 \leq k_i \leq 20{,}000$). There will be no two keys with the same k_i.

The last test case is followed by a single line containing 2 zeroes.

Output

For each test case, if it is impossible to display number r print "`Stupid keypad!`" If Mr. Ed can display r, print n integers: the i-th integer represents how many times you need to add (if the number is positive) or subtract (if the number is negative) number k_i in order to display r. See format below for details.

Please notice Mr. Ed can press a single key as many times as he like, possibly leading to multiple correct solutions. Any correct solution you print will be accepted.

Example

Input	Output

3 7
1 3 5
1 3
6
0 0
	Case #1: -1 1 1
	Case #2: Stupid keypad!