Escape

SEASON 10 - SECOND ROUND

Georgi is currently somewhere in Hotel “Trakia” in Pazardzhik and must go out to go to the supermarket. At the same time, Emil is just entering the hotel and heading upstairs. Georgi and Emil are not exactly best friends and since Georgi is very shy and wants to get out of the hotel fast, he doesn’t want to meet Emil on the way down to the exit.

There are **N** different locations in the hotel, between which our two heroes can walk - they can be rooms, lobbies, etc. There are **M** connections between them - parts of stairways and hallways. All M connections are exactly 1 meter long. Georgi doesn’t know where Emil is going, but he knows Emil travels a distance of 1 meter for exactly **a** seconds. Georgi also knows his own speed - he travels 1 meter in exactly **b** seconds. Speeds of both of them are not affected by whether they’re going through a hallway, upstairs or downstairs. Yes, we know that normally people move with greater speed than 1 m/s, but in this statement the action takes place vary late at night and both our heroes are exhausted so they move slowly. Anyway, Georgi needs such a route, that he surely won’t meet Emil at any point on his way out. There are **K** exits in the hotel, situated at locations $\mathbf{e_1}$, $\mathbf{e_2}$, ... $\mathbf{e_K}$, and Emil is entering through the first exit, i.e. location $\mathbf{e_1}$ (obviously exits are entrances too). Help Georgi escape the hotel!

Input (escape.in)

You’ll have to solve several different scenarios in a single test. On the first line you are given the number of scenarios **T**. Every scenario is described in the following way: on the first line you are given 5 integer numbers **N**, **M**, **a**, **b** and **s** - the number of locations, number of connections, times (in seconds) needed by Emil and Georgi respectively to walk 1 meter and Georgi’s starting location. On each of the next M lines you are given two numbers - indices of locations, between which the corresponding connection is situated. On the next line stands the integer **K** - the number of exits. On the last line you are given K numbers - indices of locations of the exits.

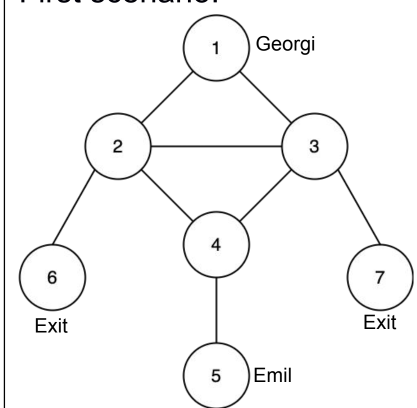
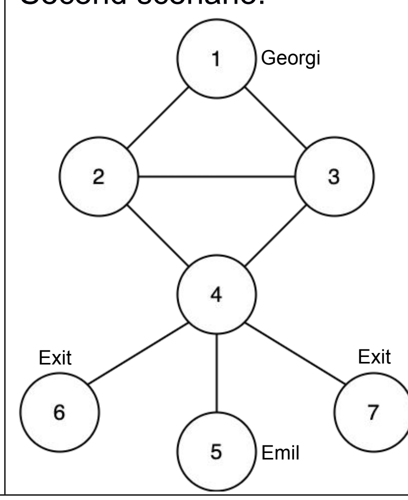
Output (escape.out)

For every scenario output the following: if Georgi can’t escape the hotel in a way that he’s sure he won’t meet Emil, print the number **-1** on a single line. Otherwise, on the first line print a single number **L** - number of locations in Georgi’s route. On the second line print L **different** indices of locations, describing the route. The first one must be Georgi’s starting location and the last one must be an exit. There must be a direct connection between every two consecutive locations. If there are several possible routes, describe any of them.

Constraints

1 ≤ N, M ≤ 1 000 000
1 ≤ a, b ≤ 10^9
1 ≤ K, s, e_i, L ≤ N
1 ≤ T ≤ 10
The sum of N, M over all scenarios ≤ 1 000 000

Example

Input	Output	Explanation
2 7 8 3 2 1 1 2 1 3 2 3 2 4 3 4 4 5 2 6 3 7 3 5 6 7 7 8 3 2 1 1 2 1 3 2 3 2 4 3 4 4 5 4 6 4 7 3 5 6 7	3 1 2 6 -1	First scenario:  Second scenario: 

Connect 'Em

SEASON 10 – SECOND ROUND

Misho loves to connect squares – this is his favorite activity to tackle boredom. He has a sheet of paper divided into $\boldsymbol{N}$ rows and $\boldsymbol{M}$ columns, numbered respectively with the numbers from 1 to N and from 1 to M. In the beginning, he places his pencil in the centre of the square with coordinates $(1, 1)$. After that, he can move it to the centre of $(1, 2)$ or $(2, 1)$. Generally, if the tip of the pencil is now placed in the cell on the i-th row and the j-th column, Misho may move it to $(i, j + 1)$ or $(i + 1, j)$. His aim is to reach the square on the last row and the last column (this with coordinates (N, M)), without lifting the pencil.

Once when he was particularly bored in class, Misho began to connect squares. His teacher noticed that and got angry. In order not to punish him, she gave him a list of $\boldsymbol{K}$ different squares and asked him to connect the cells in such a way so that the pencil would cross each of them. Unfortunately, this is not always possible. Write a program which finds the maximal number of the given squares, which the pencil could cross, as well as how many different ways to do that exist. Two ways are considered different if one of them crosses given square, while the other does not. Your program has to process $\boldsymbol{T}$ test cases during a single execution.

Input

The first line of the input file `connectem.in` contains one number T. The descriptions of T test cases follow.

On the first line of the description of each test case are written three numbers N, M и K. On each of the following K lines are written two numbers R_i and C_i – the coordinates of the cells which the teacher has given. Among them always will be $(1, 1)$ and (N, M).

Output

For each test case on a single line of the output file `connectem.out` print two numbers, separated by a space – the maximal number of squares which could be crossed and the number of ways to achieve that.

Constraints

$1 < N, M \leq 100\ 000$

$1 < K \leq 5\ 000$

$1 \leq R_i \leq N$

$1 \leq C_i \leq M$

The sum of the values of K over all test cases will not exceed 5 000.

Example

Input	Output
1 4 5 5 1 1 1 2 2 4 4 3 4 5	4 13

Monyo

SEASON 10 - SECOND ROUND

Today Monyo the milkman is going to load the trunk of his car with milk and go sell it in the villages nearby. There are **N** houses living in those villages and N-1 bidirectional roads connecting them. There is a path consisting of some of those roads between every pair of houses. The houses are numbered from 1 to N such that Monyo's house is number one. There is one potential customer living in every other house. Monyo has two types of customers: loyal and not so loyal. A loyal customer will definitely buy one bottle of milk the first time Monyo passes by their house. "Not so loyal" customers buy milk only if Monyo tries to sell them, but they also don't refuse to buy, because everyone has heard of how delicious his milk is. Loyal customers are useful, because they recommend Monyo's milk to others and thus help him expand his business. That's why he always wants to attract more customers and have more loyal ones. In order to do this, he wants to sell as much milk as possible to people who are not loyal customers yet, even if he has to avoid loyal ones (after all, they all love him and won't get angry at him for this), but he must still sell to loyal customers if he passes by their houses.

Monyo will proceed the following way: he starts from his own house and every minute he goes to another one, which is connected by a road to the one he's currently in. If he hasn't visited that house before, he will sell one bottle of milk to the person who lives there (if it's a loyal customer, Monyo is obliged to sell to him, otherwise Monyo will sell to him, because he wants to attract him as a loyal customer). If he has visited the house before, he won't do anything there. When he has no milk left, he will go home. Help him find the maximum number of "not so loyal" customers he can sell to, depending on how many bottles of milk he has.

Input (monyo.in)

On the first line of the input you are given the number of houses **N**. Each of the next N - 1 lines contains two numbers **A** and **B** - the indices of two houses connected by a road. The last line contains a string of length N-1, consisting only of zeros and ones. If the i-th character of the string is 1, the person living in house with number i+1 is a loyal customer, otherwise not.

Output (monyo.out)

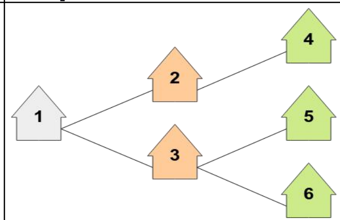
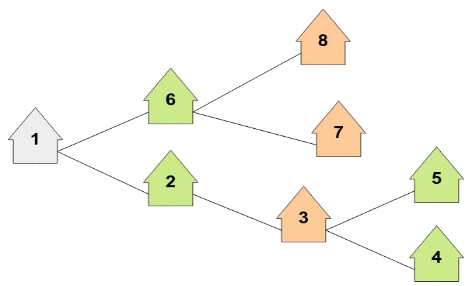
Output N-1 numbers on separate lines: the i-th number should be equal to the maximum number of "not so loyal" customers Monyo can sell milk to, if he has loaded i bottles of milk in his trunk.

Constraints

$1 \le N \le 1\ 000\ 000$

The number of loyal customers is not greater than 1 000.

Examples

Input	Output	Explanation
6 1 2 1 3 2 4 3 5 3 6 11000	0 1 2 2 3	 On the picture Monyo's house is in gray, loyal customers' houses are in orange and other customers' houses are in green. If Monyo has only one bottle, he has to sell it either in house 2 or in house 3. If he has 2 bottles, he can sell them for example in houses 2 and 4. If he has 3 bottles, he will sell them in houses 3, 5 and 6. If he has 4 bottles, he has to sell two of them in houses 2 and 3 and there will be only two left for not loyal customers. If he has 5 bottles, he will sell one in each house.
8 1 2 2 3 3 4 3 5 1 6 6 7 6 8 0100011	1 2 2 3 4 4 4	 If Monyo has 1 or 2 bottles, he can sell them in houses 2 and 6. If he has 3, 4 or 5 bottles, he can sell them by visiting only one loyal customer (the one in house 3). If he has 6 or 7 bottles, he will sell the last ones in houses 7 and 8.

Network visibility

SESON 10 – SECOND ROUND

Svetlyo has decided to study computer networks. A computer network is a set of N devices (numbered with the numbers from 1 to N), which are connected by $N-1$ cables. The network has a hierarchical structure – each device except for one (which we call main device) has a direct parent, from which it obtains an ip address.

During his activities he faced the problem to determine how many devices in the network are visible from a given device. We say that a device v is visible from other device u, if and only if one of the following conditions holds:

1) u is the parent of v in the hierarchical network
2) v is a predecessor (indirect parent) of u in the hierarchical network
3) the parent of v is a predecessor of u in the hierarchical network

Svetlyo has recently discovered a novel functionality of the network devices – the operation mode "access point" (AP). When a device is in this mode, it works only as a physical connection between the other devices and is practically invisible for them. Moreover, the devices, which it was the parent of, will now directly establish a connection to the closest their predecessor, which is not in AP, to obtain an ip address. Of course, the main device cannot be in AP mode.

In order to continue to study the network properties, Svetlyo needs a program which can answer Q queries of the following types:

1) ***T x*** – changes the mode of the device with number x (if AP is turned on – it will be turned off, and vice versa)
2) ***B x*** – the device with number x breaks down (or if it is already broken then it gets replaced with a new one). In case of such a failure the network disintegrates, and new subnetworks are formed, each of which has a main device among those, whose parent was x. If some of these main devices are in AP mode, then it is disabled. Likewise, when a device is replaced its children reconnect to it. Both the device and its children will not be in AP (even if they had been before the failure).
3) ***C x*** – counts the number of the devices, which are visible form the device with number x in the network. The device in question will never be in AP mode.

Input

On the first line of the input file `visibility.in` are written the numbers N and Q. The second line contains N numbers, the i-th of which is the number of the parent of the device with number i or 0 if it is the main device. Each of the following Q lines describes one query in the abovementioned format.

Output

For each query of third type on a single line of the output file `visibility.out` print one number – the number of the visible devices.

Constraints

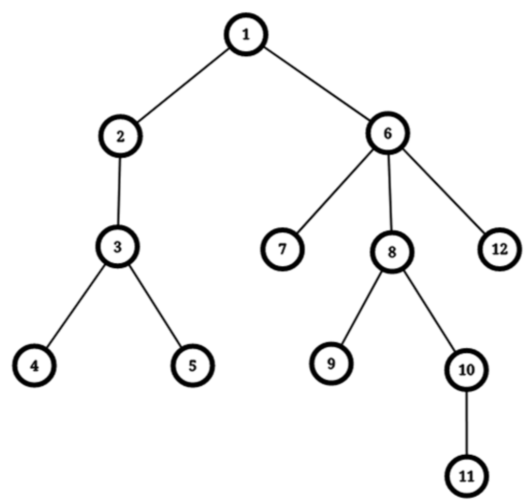
$1 < N, Q \leq 100\,000$
$1 \leq x \leq N$

Example

Input	Output
12 10 0 1 2 3 3 1 6 6 8 8 10 6 C 10 C 1 C 2 T 6 C 2 C 9 T 3 C 2 B 1 C 8	8 2 3 5 6 6 5

Explanation

The initial state of the network is represented by the picture on right.



After the fourth query, device 6 becomes invisible and 7, 8 and 12 connect directly to 1. The last three become visible for device 2.

After the seventh query, device 3 becomes invisible and 4 and 5 connect directly to 2. That is why, from device 2 there are 6 visible devices, and they are 1, 4, 5, 7, 8 and 12.

After the ninth query, device 1 fails and two new networks are formed, whose main devices will be 2 and 6, respectively. The device with number 6 will no more be in AP. In this situation, the devices visible from 8 are 6, 7, 9, 10 and 12.

Distillation

SEASON 10 - SECOND ROUND

Raya appreciates good whiskey very much. She has gathered information about **N** small distilleries, numbered from 1 to N, and wants to invest money in one of them. The process of making whiskey is complicated, but we can consider it simply in two phases:

1. First the barley and malt are roasted and left to ferment to obtain a liquor, which is then distilled and turned into whiskey.

2. After that the whiskey is left to age - the longer it ages, the better its quality is at the end of the process.

The second phase can't begin before the first one is finished, i.e. first some amount of whiskey is produced, then it's left in barrels to age and no more whiskey is produced after that.

Raya is looking very strictly after the quality of the whiskey, therefore she measures it with numbers. For the i-th distillery she has written down 4 real numbers $\mathbf{A_i}$, $\mathbf{B_i}$, $\mathbf{C_i}$ and $\mathbf{D_i}$, for which the following conditions hold:

1. The distillery need A_i days for maintenance before it can start producing whiskey.

2. During the first phase the distillery will produce B_i liters of whiskey with quality C_i every day.

3. During the second phase the quality of the whiskey will increase with D_i units per day. The quality of the whiskey doesn't change before the start of the second phase.

Raya wants you to help her decide in which distillery to invest by answering **Q** questions of the following type: What is the maximum value of quantity (in liters) multiplied by quality, that one distillery can achieve for **T** days, if every distillery allocates the time optimally for itself, and which is the distillery that achieves the best result? Please note that T is not necessarily an integer and that both phases of making whiskey proceed evenly, i.e. if a distillery produces B liters per day, it will produce for example $\frac{B}{2}$ liters for half a day and if the quality of the whiskey improves with D units per day, it will improve with $\frac{D}{2}$ units for half a day.

Input (distillation.in)

The first line of the input contains the integer **N** - the number of distilleries. Each of the following N lines contain 4 real numbers $\mathbf{A_i}$, $\mathbf{B_i}$, $\mathbf{C_i}$ and $\mathbf{D_i}$ - the number Raya has written down for the i-th distillery. The next line contains the integer **Q** - the number of questions you have to answer. Each of the following Q lines contains a number $\mathbf{T_j}$ - the number of days for the j-th question.

Output (distillation.out)

For each question output on a separate line two numbers, separated by a single space. The first number should be equal to the greatest possible product of quality and quantity one distillery can achieve and the second number should be the index of the distillery that can achieve this product. If there are two or more such distilleries, output the smallest index of one of them. The first number will be considered correct if the absolute difference between it and the author's answer is not greater than 10^{-5}.

Constraints

$1 \le N, Q \le 100\,000$

$0 \le A_i, C_i \le 1\,000$

$0 < B_i, D_i, T_j \le 1\,000$

A_i, B_i, C_i, D_i and T_j are given with no more than 5 digits after the decimal point.

Example

Input

2
3 2.5 1.5 2
1 1.25 0 1.25
3
2
3.5
6

Output

0.390625 2
2.441406 2
17.578125 1