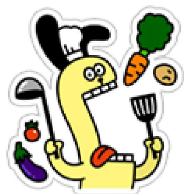
Lista AA 2022 #9

A. Tavas and Karafs

2 seconds, 256 megabytes

Karafs is some kind of vegetable in shape of an $1 \times h$ rectangle. Tavaspolis people love Karafs and they use Karafs in almost any kind of food. Tavas, himself, is crazy about Karafs.



Each Karafs has a positive integer height. Tavas has an infinite **1-based** sequence of Karafses. The height of the i-th Karafs is $s_i = A + (i - 1) \times B$.

For a given m, let's define an m-bite operation as decreasing the height of at most m distinct not eaten Karafses by 1. Karafs is considered as eaten when its height becomes zero.

Now SaDDas asks you n queries. In each query he gives you numbers l, t and m and you should find the largest number r such that $l \le r$ and sequence $s_l, s_{l+1}, ..., s_r$ can be eaten **by performing m-bite no more than t times** or print -1 if there is no such number r.

Input

The first line of input contains three integers A, B and n ($1 \le A, B \le 10^6$, $1 \le n \le 10^5$).

Next n lines contain information about queries. i-th line contains integers l, t, m ($1 \le l, t, m \le 10^6$) for i-th query.

Output

For each query, print its answer in a single line.

input
```
2 1 4
1 5 3
3 3 10
7 10 2
6 4 8
```
output
```
4
-1
8
-1
```

input
```
1 5 2
1 5 10
2 7 4
```
output
```
1
2
```

B. Hanoi Factory

1 second, 256 megabytes

Of course you have heard the famous task about Hanoi Towers, but did you know that there is a special factory producing the rings for this wonderful game? Once upon a time, the ruler of the ancient Egypt ordered the workers of Hanoi Factory to create as high tower as possible. They were not ready to serve such a strange order so they had to create this new tower using already produced rings.

There are n rings in factory's stock. The i-th ring has inner radius a_i, outer radius b_i and height h_i. The goal is to select some subset of rings and arrange them such that the following conditions are satisfied:

- Outer radiuses form a non-increasing sequence, i.e. one can put the j-th ring on the i-th ring only if $b_j \le b_i$.
- Rings should not fall one into the the other. That means one can place ring j on the ring i only if $b_j > a_i$.
- The total height of all rings used should be maximum possible.

Input

The first line of the input contains a single integer n ($1 \le n \le 100\,000$) — the number of rings in factory's stock.

The i-th of the next n lines contains three integers a_i, b_i and h_i ($1 \le a_i, b_i, h_i \le 10^9$, $b_i > a_i$) — inner radius, outer radius and the height of the i-th ring respectively.

Output

Print one integer — the maximum height of the tower that can be obtained.

input
```
3
1 5 1
2 6 2
3 7 3
```
output
```
6
```

input
```
4
1 2 1
1 3 3
4 6 2
5 7 1
```
output
```
4
```

In the first sample, the optimal solution is to take all the rings and put them on each other in order $3, 2, 1$.

In the second sample, one can put the ring 3 on the ring 4 and get the tower of height 3, or put the ring 1 on the ring 2 and get the tower of height 4.

C. DZY Loves Modification

2 seconds, 256 megabytes

As we know, DZY loves playing games. One day DZY decided to play with a $n \times m$ matrix. To be more precise, he decided to modify the matrix with exactly k operations.

Each modification is one of the following:

1. Pick some row of the matrix and decrease each element of the row by p. This operation brings to DZY the value of pleasure equal to the sum of elements of the row before the decreasing.
2. Pick some column of the matrix and decrease each element of the column by p. This operation brings to DZY the value of pleasure equal to the sum of elements of the column before the decreasing.

DZY wants to know: what is the largest total value of pleasure he could get after performing exactly k modifications? Please, help him to calculate this value.

Input

The first line contains four space-separated integers n, m, k and p $(1 \le n, m \le 10^3;\ 1 \le k \le 10^6;\ 1 \le p \le 100)$.

Then n lines follow. Each of them contains m integers representing a_{ij} $(1 \le a_{ij} \le 10^3)$ — the elements of the current row of the matrix.

Output

Output a single integer — the maximum possible total pleasure value DZY could get.

input
```
2 2 2 2
1 3
2 4
```
output
```
11
```

input
```
2 2 5 2
1 3
2 4
```
output
```
11
```

For the first sample test, we can modify: column 2, row 2. After that the matrix becomes:

```
1 1
0 0
```

For the second sample test, we can modify: column 2, row 2, row 1, column 1, column 2. After that the matrix becomes:

```
-3 -3
-2 -2
```

D. Image Preview

1 second, 256 megabytes

Vasya's telephone contains n photos. Photo number 1 is currently opened on the phone. It is allowed to move left and right to the adjacent photo by swiping finger over the screen. If you swipe left from the first photo, you reach photo n. Similarly, by swiping right from the last photo you reach photo 1. It takes a seconds to swipe from photo to adjacent.

For each photo it is known which orientation is intended for it — horizontal or vertical. Phone is in the vertical orientation and **can't** be rotated. It takes b second to change orientation of the photo.

Vasya has T seconds to watch photos. He want to watch as many photos as possible. If Vasya opens the photo for the first time, he spends 1 second to notice all details in it. If photo is in the wrong orientation, he spends b seconds on rotating it before watching it. If Vasya has already opened the photo, he just skips it (so he doesn't spend any time for watching it or for changing its orientation). It is not allowed to skip unseen photos.

Help Vasya find the maximum number of photos he is able to watch during T seconds.

Input

The first line of the input contains 4 integers n, a, b, T $(1 \le n \le 5 \cdot 10^5, 1 \le a, b \le 1000, 1 \le T \le 10^9)$ — the number of photos, time to move from a photo to adjacent, time to change orientation of a photo and time Vasya can spend for watching photo.

Second line of the input contains a string of length n containing symbols 'w' and 'h'.

If the i-th position of a string contains 'w', then the photo i should be seen in the **horizontal** orientation.

If the i-th position of a string contains 'h', then the photo i should be seen in **vertical** orientation.

Output

Output the only integer, the maximum number of photos Vasya is able to watch during those T seconds.

input
```
4 2 3 10
wwhw
```
output
```
2
```

input
```
5 2 4 13
hhwhh
```
output
```
4
```

input
```
5 2 4 1000
hhwhh
```
output
```
5
```

input
```
3 1 100 10
whw
```
output
```
0
```

In the first sample test you can rotate the first photo (3 seconds), watch the first photo (1 seconds), move left (2 second), rotate fourth photo (3 seconds), watch fourth photo (1 second). The whole process takes exactly 10 seconds.

Note that in the last sample test the time is not enough even to watch the first photo, also you can't skip it.

E. Petya and Divisors

5 seconds, 256 megabytes

Little Petya loves looking for numbers' divisors. One day Petya came across the following problem:

You are given n queries in the form "x_i y_i". For each query Petya should count how many divisors of number x_i divide none of the numbers $x_{i-y_i}, x_{i-y_i+1}, ..., x_{i-1}$. Help him.

Input

The first line contains an integer n $(1 \le n \le 10^5)$. Each of the following n lines contain two space-separated integers x_i and y_i $(1 \le x_i \le 10^5$, $0 \le y_i \le i - 1$, where i is the query's ordinal number; the numeration starts with $1)$.

If $y_i = 0$ for the query, then the answer to the query will be the number of divisors of the number x_i. In this case you do not need to take the previous numbers x into consideration.

Output

For each query print the answer on a single line: the number of positive integers k such that $x_i \ mod \ k = 0 \ \& \ (\forall j : i - y_i \le j < i) \ x_j \ mod \ k \ne 0$

input

```
6
4 0
3 1
5 2
6 2
18 4
10000 3
```

output

```
3
1
1
2
2
22
```

Let's write out the divisors that give answers for the first 5 queries:

1) 1, 2, 4

2) 3

3) 5

4) 2, 6

5) 9, 18

F. Ilya And The Tree

2 seconds, 256 megabytes

Ilya is very fond of graphs, especially trees. During his last trip to the forest Ilya found a very interesting tree rooted at vertex 1. There is an integer number written on each vertex of the tree; the number written on vertex i is equal to a_i.

Ilya believes that the beauty of the vertex x is the greatest common divisor of all numbers written on the vertices on the path from the root to x, including this vertex itself. In addition, Ilya can change the number in one arbitrary vertex to 0 or leave all vertices unchanged. Now for each vertex Ilya wants to know the maximum possible beauty it can have.

For each vertex the answer must be considered independently.

The beauty of the root equals to number written on it.

Input

First line contains one integer number n — the number of vertices in tree $(1 \le n \le 2 \cdot 10^5)$.

Next line contains n integer numbers a_i $(1 \le i \le n, 1 \le a_i \le 2 \cdot 10^5)$.

Each of next $n - 1$ lines contains two integer numbers x and y $(1 \le x, y \le n, x \ne y)$, which means that there is an edge (x, y) in the tree.

Output

Output n numbers separated by spaces, where i-th number equals to maximum possible beauty of vertex i.

input

```
2
6 2
1 2
```

output

```
6 6
```

input

```
3
6 2 3
1 2
1 3
```

output

```
6 6 6
```

input

```
1
10
```

output

```
10
```

G. Masha-forgetful

3 seconds, 256 megabytes

Masha meets a new friend and learns his phone number — s. She wants to remember it as soon as possible. The phone number — is a string of length m that consists of digits from 0 to 9. The phone number may start with `0`.

Masha already knows n phone numbers (all numbers have the same length m). It will be easier for her to remember a new number if the s is represented as segments of numbers she already knows. Each such segment must be of length **at least** 2, otherwise there will be too many segments and Masha will get confused.

For example, Masha needs to remember the number: $s =$ '`12345678`' and she already knows $n = 4$ numbers: '`12340219`', '`20215601`', '`56782022`', '`12300678`'. You can represent s as a 3 segment: '`1234`' of number one, '`56`' of number two, and '`78`' of number three. There are other ways to represent s.

Masha asks you for help, she asks you to break the string s into segments of length 2 or more of the numbers she already knows. If there are several possible answers, print **any** of them.

Input

The first line of input data contains an integer t $(1 \le t \le 10^4)$ —the number of test cases.

Before each test case there is a blank line. Then there is a line containing integers n and m $(1 \le n, m \le 10^3)$ —the number of phone numbers that Masha knows and the number of digits in each phone number. Then follow n line, i-th of which describes the i-th number that Masha knows. The next line contains the phone number of her new friend s.

Among the given $n+1$ phones, there may be duplicates (identical phones).

It is guaranteed that the sum of $n \cdot m$ (n multiplied by m) values over all input test cases does not exceed 10^6.

Output

You need to print the answers to t test cases. The first line of the answer should contain one number k, corresponding to the number of segments into which you split the phone number s. Print `-1` if you cannot get such a split.

If the answer is yes, then follow k lines containing triples of numbers l, r, i. Such triplets mean that the next $r-l+1$ digits of number s are equal to a segment (substring) with boundaries $[l, r]$ of the phone under number i. Both the phones and the digits in them are numbered from 1. Note that $r-l+1 \ge 2$ for all k lines.

input

```
5

4 8
12340219
20215601
56782022
12300678
12345678

2 3
134
126
123

1 4
1210
1221

4 3
251
064
859
957
054

4 7
7968636
9486033
4614224
5454197
9482268
```

output

```
3
1 4 1
5 6 2
3 4 3
-1
2
1 2 1
2 3 1
-1
3
1 3 2
5 6 3
3 4 1
```

The example from the statement.

In the second case, it is impossible to represent by segments of known numbers of length 2 or more.

In the third case, you can get the segments '`12`' and '`21`' from the first phone number.

H. Kay and Snowflake

3 seconds, 256 megabytes

After the piece of a devilish mirror hit the Kay's eye, he is no longer interested in the beauty of the roses. Now he likes to watch snowflakes.

Once upon a time, he found a huge snowflake that has a form of the tree (connected acyclic graph) consisting of n nodes. The root of tree has index 1. Kay is very interested in the structure of this tree.

After doing some research he formed q queries he is interested in. The i-th query asks to find a centroid of the subtree of the node v_i. Your goal is to answer all queries.

Subtree of a node is a part of tree consisting of this node and all it's descendants (direct or not). In other words, subtree of node v is formed by nodes u, such that node v is present on the path from u to root.

Centroid of a tree (or a subtree) is a node, such that if we erase it from the tree, the maximum size of the connected component will be at least two times smaller than the size of the initial tree (or a subtree).

Input

The first line of the input contains two integers n and q ($2 \le n \le 300\,000$, $1 \le q \le 300\,000$) — the size of the initial tree and the number of queries respectively.

The second line contains n - 1 integer $p_2, p_3, ..., p_n$ ($1 \le p_i \le n$) — the indices of the parents of the nodes from 2 to n. Node 1 is a root of the tree. It's guaranteed that p_i define a correct tree.

Each of the following q lines contain a single integer v_i ($1 \le v_i \le n$) — the index of the node, that define the subtree, for which we want to find a centroid.

Output

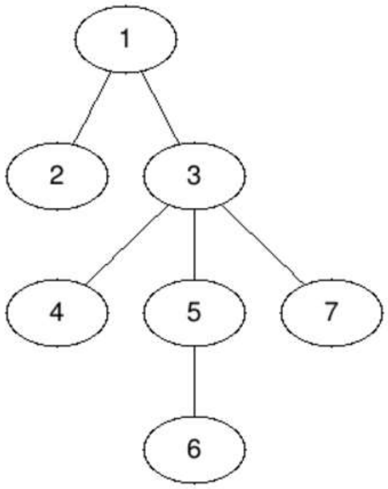
For each query print the index of a centroid of the corresponding subtree. If there are many suitable nodes, print any of them. It's guaranteed, that each subtree has at least one centroid.

input

```
7 4
1 1 3 3 5 3
1
2
3
5
```

output

```
3
2
3
6
```



The first query asks for a centroid of the whole tree — this is node 3. If we delete node 3 the tree will split in four components, two of size 1 and two of size 2.

The subtree of the second node consists of this node only, so the answer is 2.

Node 3 is centroid of its own subtree.

The centroids of the subtree of the node 5 are nodes 5 and 6 — both answers are considered correct.

I. Number With The Given Amount Of Divisors

2 seconds, 256 megabytes

Given the number n, find the smallest positive integer which has exactly n divisors. It is guaranteed that for the given n the answer will not exceed 10^{18}.

Input

The first line of the input contains integer n ($1 \le n \le 1000$).

Output

Output the smallest positive integer with exactly n divisors.

input
```
4
```
output
```
6
```

input
```
6
```
output
```
12
```

J. Missile Silos

2 seconds, 256 megabytes

A country called Berland consists of n cities, numbered with integer numbers from 1 to n. Some of them are connected by bidirectional roads. Each road has some length. There is a path from each city to any other one by these roads. According to some Super Duper Documents, Berland is protected by the Super Duper Missiles. The exact position of the Super Duper Secret Missile Silos is kept secret but Bob managed to get hold of the information. That information says that all silos are located exactly at a distance l from the capital. The capital is located in the city with number s.

The documents give the formal definition: the Super Duper Secret Missile Silo is located at some place (which is either city or a point on a road) if and only if the shortest distance from this place to the capital along the roads of the country equals exactly l.

Bob wants to know how many missile silos are located in Berland to sell the information then to enemy spies. Help Bob.

Input

The first line contains three integers n, m and s ($2 \le n \le 10^5$, $n - 1 \le m \le \min(10^5, \frac{n(n-1)}{2})$, $1 \le s \le n$) — the number of cities, the number of roads in the country and the number of the capital, correspondingly. Capital is the city no. s.

Then m lines contain the descriptions of roads. Each of them is described by three integers v_i, u_i, w_i ($1 \le v_i, u_i \le n$, $v_i \ne u_i$, $1 \le w_i \le 1000$), where v_i, u_i are numbers of the cities connected by this road and w_i is its length. The last input line contains integer l ($0 \le l \le 10^9$) — the distance from the capital to the missile silos. It is guaranteed that:

- between any two cities no more than one road exists;
- each road connects two different cities;
- from each city there is at least one way to any other city by the roads.

Output

Print the single number — the number of Super Duper Secret Missile Silos that are located in Berland.

input
```
4 6 1
1 2 1
1 3 3
2 3 1
2 4 1
3 4 1
1 4 2
2
```
output
```
3
```

input
```
5 6 3
3 1 1
3 2 1
3 4 1
3 5 1
1 2 6
4 5 8
4
```
output
```
3
```

In the first sample the silos are located in cities 3 and 4 and on road $(1, 3)$ at a distance 2 from city 1 (correspondingly, at a distance 1 from city 3).

In the second sample one missile silo is located right in the middle of the road $(1, 2)$. Two more silos are on the road $(4, 5)$ at a distance 3 from city 4 in the direction to city 5 and at a distance 3 from city 5 to city 4.