Mail.Ru Cup 2018 Round 3

A. Determine Line

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Arkady's morning seemed to be straight of his nightmare. He overslept through the whole morning and, still half-asleep, got into the tram that arrived the first. Some time after, leaving the tram, he realized that he was not sure about the line number of the tram he was in.

During his ride, Arkady woke up several times and each time he saw the tram stopping at some stop. For each stop he knows which lines of tram stop there. Given this information, can you help Arkady determine what are the possible lines of the tram he was in?

Input

The first line contains a single integer n ($2 \le n \le 100$) — the number of stops Arkady saw.

The next n lines describe the stops. Each of them starts with a single integer r ($1 \le r \le 100$) — the number of tram lines that stop there. r distinct integers follow, each one between 1 and 100, inclusive, — the line numbers. They can be in arbitrary order.

It is guaranteed that Arkady's information is consistent, i.e. there is at least one tram line that Arkady could take.

Output

Print all tram lines that Arkady could be in, in arbitrary order.

Examples

input
3 3 1 4 6 2 1 4 5 10 5 6 4 1
output
1 4

input
5 1 1 10 10 9 8 7 100 5 4 3 99 1 5 1 2 3 4 5 5 4 1 3 2 5 4 10 1 5 3
output
1

Note

Consider the first example. Arkady woke up three times. The first time he saw a stop with lines 1, 4, 6. The second time he saw a stop with lines 1, 4. The third time he saw a stop with lines 10, 5, 6, 4 and 1. He can be in a tram of one of two lines: 1 or 4.

B. Divide Candies

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Arkady and his friends love playing checkers on an $n \times n$ field. The rows and the columns of the field are enumerated from 1 to n.

The friends have recently won a championship, so Arkady wants to please them with some candies. Remembering an old parable (but not its moral), Arkady wants to give to his friends one set of candies per each cell of the field: the set of candies for cell (i, j) will have exactly $(i^2 + j^2)$ candies of unique type.

There are m friends who deserve the present. How many of these $n \times n$ sets of candies can be split equally into m parts without cutting a candy into pieces? Note that each set has to be split independently since the types of candies in different sets are different.

Input

The only line contains two integers n and m ($1 \le n \le 10^9$, $1 \le m \le 1000$) — the size of the field and the number of parts to split

the sets into.

Output
Print a single integer — the number of sets that can be split equally.

Examples

input
3 3
output
1

input
6 5
output
13

input
1000000000 1
output
1000000000000000000

Note
In the first example, only the set for cell $(3, 3)$ can be split equally ($3^2 + 3^2 = 18$, which is divisible by $m = 3$).

In the second example, the sets for the following cells can be divided equally:

- $(1, 2)$ and $(2, 1)$, since $1^2 + 2^2 = 5$, which is divisible by 5;
- $(1, 3)$ and $(3, 1)$;
- $(2, 4)$ and $(4, 2)$;
- $(2, 6)$ and $(6, 2)$;
- $(3, 4)$ and $(4, 3)$;
- $(3, 6)$ and $(6, 3)$;
- $(5, 5)$.

In the third example, sets in all cells can be divided equally, since $m = 1$.

C. Pick Heroes

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Don't you tell me what you think that I can be

If you say that Arkady is a bit old-fashioned playing checkers, you won't be right. There is also a modern computer game Arkady and his friends are keen on. We won't discuss its rules, the only feature important to this problem is that each player has to pick a distinct hero in the beginning of the game.

There are 2 teams each having n players and $2n$ heroes to distribute between the teams. The teams take turns picking heroes: at first, the first team chooses a hero in its team, after that the second team chooses a hero and so on. Note that after a hero is chosen it becomes unavailable to both teams.

The friends estimate the power of the i-th of the heroes as p_i. Each team wants to maximize the total power of its heroes. However, there is one exception: there are m pairs of heroes that are especially strong against each other, so when any team chooses a hero from such a pair, the other team **must** choose the other one on its turn. Each hero is in at most one such pair.

This is an interactive problem. You are to write a program that will optimally choose the heroes for one team, while the jury's program will play for the other team. Note that the jury's program may behave inefficiently, in this case you have to take the opportunity and still maximize the total power of your team. Formally, if you ever have chance to reach the total power of q or greater regardless of jury's program choices, you must get q or greater to pass a test.

Input
The first line contains two integers n and m ($1 \le n \le 10^3$, $0 \le m \le n$) — the number of players in one team and the number of special pairs of heroes.

The second line contains $2n$ integers $p_1, p_2, \ldots, p_{2n}$ ($1 \le p_i \le 10^3$) — the powers of the heroes.

Each of the next m lines contains two integer a and b ($1 \le a, b \le 2n$, $a \ne b$) — a pair of heroes that are especially strong against each other. It is guaranteed that each hero appears at most once in this list.

The next line contains a single integer t ($1 \le t \le 2$) — the team you are to play for. If $t = 1$, the first turn is yours, otherwise you have the second turn.

Hacks

In order to hack, use the format described above with one additional line. In this line output $2n$ distinct integers from 1 to $2n$ — the priority order for the jury's team. The jury's team will on each turn select the first possible hero from this list. Here possible means that it is not yet taken and does not contradict the rules about special pair of heroes.

Interaction

When it is your turn, print a single integer x ($1 \le x \le 2n$) — the index of the hero chosen by you. Note that you can't choose a hero previously chosen by either you of the other player, and you must follow the rules about special pairs of heroes.

When it is the other team's turn, read a line containing a single integer x ($1 \le x \le 2n$) — the index of the hero chosen by the other team. It is guaranteed that this index is not chosen before and that the other team also follows the rules about special pairs of heroes.

After the last turn you should terminate without printing anything.

After printing your choice do not forget to output end of line and flush the output. Otherwise you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

Jury's answer -1 instead of a valid choice means that you made an invalid turn. Exit immediately after receiving -1 and you will see `Wrong answer` verdict. Otherwise you can get an arbitrary verdict because your solution will continue to read from a closed stream.

Examples

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

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

Note

In the first example the first turn is yours. In example, you choose 6, the other team is forced to reply with 2. You choose 5, the other team chooses 4. Finally, you choose 3 and the other team choose 1.

In the second example you have the second turn. The other team chooses 6, you choose 5, forcing the other team to choose 1. Now you choose 4, the other team chooses 3 and you choose 2.

D. Decorate Apple Tree

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

There is one apple tree in Arkady's garden. It can be represented as a set of junctions connected with branches so that there is only one way to reach any junctions from any other one using branches. The junctions are enumerated from 1 to n, the junction 1 is called the root.

A subtree of a junction v is a set of junctions u such that the path from u to the root must pass through v. Note that v itself is included in a subtree of v.

A leaf is such a junction that its subtree contains exactly one junction.

The New Year is coming, so Arkady wants to decorate the tree. He will put a light bulb of some color on each leaf junction and then count the number happy junctions. A happy junction is such a junction t that all light bulbs in the subtree of t have different colors.

Arkady is interested in the following question: for each k from 1 to n, what is the minimum number of different colors needed to make the number of happy junctions be greater than or equal to k?

Input

The first line contains a single integer n ($1 \le n \le 10^5$) — the number of junctions in the tree.

The second line contains $n - 1$ integers p_2, p_3, ..., p_n ($1 \le p_i < i$), where p_i means there is a branch between junctions i and p_i. It is guaranteed that this set of branches forms a tree.

Output

Output n integers. The i-th of them should be the minimum number of colors needed to make the number of happy junctions be at least i.

Examples

input
3 1 1
output
1 1 2

input
5 1 1 3 3
output
1 1 1 2 3

Note

In the first example for $k = 1$ and $k = 2$ we can use only one color: the junctions 2 and 3 will be happy. For $k = 3$ you have to put the bulbs of different colors to make all the junctions happy.

In the second example for $k = 4$ you can, for example, put the bulbs of color 1 in junctions 2 and 4, and a bulb of color 2 into junction 5. The happy junctions are the ones with indices 2, 3, 4 and 5 then.

E. Check Transcription

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

One of Arkady's friends works at a huge radio telescope. A few decades ago the telescope has sent a signal s towards a faraway galaxy. Recently they've received a response t which they believe to be a response from aliens! The scientists now want to check if the signal t is similar to s.

The original signal s was a sequence of zeros and ones (everyone knows that binary code is the universe-wide language). The returned signal t, however, does not look as easy as s, but the scientists don't give up! They represented t as a sequence of English letters and say that t is similar to s if you can replace all zeros in s with some string r_0 and all ones in s with some other string r_1 and obtain t. The strings r_0 and r_1 must be different and non-empty.

Please help Arkady's friend and find the number of possible replacements for zeros and ones (the number of pairs of strings r_0 and r_1) that transform s to t.

Input
The first line contains a string s ($2 \le |s| \le 10^5$) consisting of zeros and ones — the original signal.

The second line contains a string t ($1 \le |t| \le 10^6$) consisting of lowercase English letters only — the received signal.

It is guaranteed, that the string s contains at least one '0' and at least one '1'.

Output
Print a single integer — the number of pairs of strings r_0 and r_1 that transform s to t.

In case there are no such pairs, print 0.

Examples

input
01 aaaaaa
output
4

input
001 kokokokotlin
output
2

Note
In the first example, the possible pairs (r_0, r_1) are as follows:

- "a", "aaaaa"
- "aa", "aaaa"
- "aaaa", "aa"
- "aaaaa", "a"

The pair "aaa", "aaa" is not allowed, since r_0 and r_1 must be different.

In the second example, the following pairs are possible:

- "ko", "kokotlin"
- "koko", "tlin"

F. Write The Contest

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarp, Arkady's friend, prepares to the programming competition and decides to write a contest. The contest consists of n problems and lasts for T minutes. Each of the problems is defined by two positive integers a_i and p_i — its difficulty and the score awarded by its solution.

Polycarp's experience suggests that his skill level is defined with positive real value s, and initially $s = 1.0$. To solve the i-th problem Polycarp needs a_i/s minutes.

Polycarp loves to watch series, and before solving each of the problems he will definitely watch one episode. After Polycarp watches an episode, his skill decreases by 10%, that is skill level s decreases to $0.9s$. Each episode takes exactly 10 minutes to watch. When Polycarp decides to solve some problem, he firstly has to watch one episode, and only then he starts solving the problem without breaks for a_i/s minutes, where s is his current skill level. In calculation of a_i/s no rounding is performed, only division of integer value a_i by real value s happens.

Also, Polycarp can train for some time. If he trains for t minutes, he increases his skill by $C \cdot t$, where C is some given positive real constant. Polycarp can train only before solving any problem (and before watching series). Duration of the training can be arbitrary real value.

Polycarp is interested: what is the largest score he can get in the contest? It is allowed to solve problems in any order, while training is only allowed **before solving the first problem**.

Input
The first line contains one integer tc ($1 \le tc \le 20$) — the number of test cases. Then tc test cases follow.

The first line of each test contains one integer n ($1 \le n \le 100$) — the number of problems in the contest.

The second line of the test contains two real values C, T ($0 < C < 10$, $0 \le T \le 2 \cdot 10^5$), where C defines the efficiency of the training and T is the duration of the contest in minutes. Value C, T are given exactly with three digits after the decimal point.

Each of the next n lines of the test contain characteristics of the corresponding problem: two integers a_i, p_i ($1 \le a_i \le 10^4$, $1 \le p_i \le 10$) — the difficulty and the score of the problem.

It is guaranteed that the value of T is such that changing it by the 0.001 in any direction will not change the test answer.

Please note that in **hacks** you can only use $tc = 1$.

Output
Print tc integers — the maximum possible score in each test case.

Examples

input
2 4 1.000 31.000 12 3 20 6 30 1 5 1 3 1.000 30.000 1 10 10 10 20 8
output
7 20

Note
In the first example, Polycarp can get score of 7 as follows:

1. Firstly he trains for 4 minutes, increasing s to the value of 5;
2. Then he decides to solve 4-th problem: he watches one episode in 10 minutes, his skill level decreases to $s = 5 * 0.9 = 4.5$ and then he solves the problem in $5/s = 5/4.5$, which is roughly 1.111 minutes;
3. Finally, he decides to solve 2-nd problem: he watches one episode in 10 minutes, his skill level decreases to $s = 4.5 * 0.9 = 4.05$ and then he solves the problem in $20/s = 20/4.05$, which is roughly 4.938 minutes.

This way, Polycarp uses roughly $4 + 10 + 1.111 + 10 + 4.938 = 30.049$ minutes, to get score of 7 points. It is not possible to achieve larger score in 31 minutes.

In the second example, Polycarp can get 20 points as follows:

1. Firstly he trains for 4 minutes, increasing s to the value of 5;
2. Then he decides to solve 1-st problem: he watches one episode in 10 minutes, his skill decreases to $s = 5 * 0.9 = 4.5$ and then he solves problem in $1/s = 1/4.5$, which is roughly 0.222 minutes.
3. Finally, he decides to solve 2-nd problem: he watches one episode in 10 minutes, his skill decreases to $s = 4.5 * 0.9 = 4.05$ and then he solves the problem in $10/s = 10/4.05$, which is roughly 2.469 minutes.

This way, Polycarp gets score of 20 in $4 + 10 + 0.222 + 10 + 2.469 = 26.691$ minutes. It is not possible to achieve larger score in 30 minutes.

G. Take Metro

time limit per test: 2 seconds
memory limit per test: 1024 megabytes
input: standard input
output: standard output

Having problems with tram routes in the morning, Arkady decided to return home by metro. Fortunately for Arkady, there is only one metro line in the city.

Unfortunately for Arkady, the line is circular. It means that the stations are enumerated from 1 to n and there is a tunnel between any pair of consecutive stations as well as between the station 1 and the station n. Trains that go in clockwise direction visit the stations in order $1 \to 2 \to 3 \to \ldots \to n \to 1$ while the trains that go in the counter-clockwise direction visit the stations in the reverse order.

The stations that have numbers from 1 to m have interior mostly in red colors while the stations that have numbers from $m + 1$ to n have blue interior. Arkady entered the metro at station s and decided to use the following algorithm to choose his way home.

1. Initially he has a positive integer t in his mind.
2. If the current station has red interior, he takes a clockwise-directed train, otherwise he takes a counter-clockwise-directed train.
3. He rides exactly t stations on the train and leaves the train.
4. He decreases t by one. If t is still positive, he returns to step 2. Otherwise he exits the metro.

You have already realized that this algorithm most probably won't bring Arkady home. Find the station he will exit the metro at so that you can continue helping him.

Input
The first line contains two integers n and m ($3 \le n \le 10^5$, $1 \le m < n$) — the total number of stations and the number of stations that have red interior.

The second line contains two integers s and t ($1 \le s \le n$, $1 \le t \le 10^{12}$) — the starting station and the initial value of t.

Output
Output the only integer — the station where Arkady will exit the metro.

Examples

input
10 4 3 1
output
4

input
10 4 3 5
output
4

input
10543 437 5492 1947349
output
438

Note
Consider the first example. There are 10 stations and the first 4 of them are red. Arkady starts at station 3 with value $t = 1$, so just rides 1 station in clockwise direction and ends up on the station 4.

In the second example the metro is same as in the first example, but Arkady starts at station 3 with value $t = 5$.

- It is a red station so he rides 5 stations in clockwise direction and leaves the train at station 8.
- It is a blue station, so he rides 4 stations in counter-clockwise direction and leaves at station 4.
- It is a red station, so he rides 3 stations in clockwise direction and leaves at station 7.
- It is a blue station, so he rides 2 stations in counter-clockwise direction and leaves at station 5.
- It is a blue station, so he rides 1 station in counter-clockwise direction and leaves at station 4.

Now $t = 0$, so Arkady exits metro at the station 4.

H. Detect Robots

time limit per test: 1 second
memory limit per test: 1024 megabytes
input: standard input
output: standard output

You successfully found poor Arkady near the exit of the station you've perfectly predicted. You sent him home on a taxi and suddenly came up with a question.

There are n crossroads in your city and several bidirectional roads connecting some of them. A *taxi ride* is a path from some crossroads to another one without passing the same crossroads twice. You have a collection of rides made by one driver and now you wonder if this driver can be a robot or they are definitely a human.

You think that the driver can be a robot if for every two crossroads a and b the driver always chooses the same path whenever he drives from a to b. Note that a and b here do not have to be the endpoints of a ride and that the path from b to a can be different. On the contrary, if the driver ever has driven two different paths from a to b, they are definitely a human.

Given the system of roads and the description of all rides available to you, determine if the driver can be a robot or not.

Input
Each test contains one or more test cases. The first line contains a single integer t ($1 \le t \le 3 \cdot 10^5$) — the number of test cases.

The first line of each test case contains a single integer n ($1 \le n \le 3 \cdot 10^5$) — the number of crossroads in the city.

The next line contains a single integer q ($1 \le q \le 3 \cdot 10^5$) — the number of rides available to you.

Each of the following q lines starts with a single integer k ($2 \le k \le n$) — the number of crossroads visited by the driver on this ride. It is followed by k integers c_1, c_2, ..., c_k ($1 \le c_i \le n$) — the crossroads in the order the driver visited them. It is guaranteed that all crossroads in one ride are distinct.

It is guaranteed that the sum of values k among all rides of all test cases does not exceed $3 \cdot 10^5$.

It is guaranteed that the sum of values n and the sum of values q doesn't exceed $3 \cdot 10^5$ among all test cases.

Output

Output a single line for each test case.

If the driver can be a robot, output "Robot" in a single line. Otherwise, output "Human".

You can print each letter in any case (upper or lower).

Examples

input
1 5 2 4 1 2 3 5 3 1 4 3
output
Human

input
1 4 4 3 1 2 3 3 2 3 4 3 3 4 1 3 4 1 2
output
Robot

Note

In the first example it is clear that the driver used two different ways to get from crossroads 1 to crossroads 3. It must be a human.

In the second example the driver always drives the cycle $1 \to 2 \to 3 \to 4 \to 1$ until he reaches destination.