Problem A – Another Problem About Maximum in Range

Given a sequence of integers $a_1, a_2, \ldots, a_N$, find:

$$\left(\sum_{i=1}^{N}\sum_{j=i}^{N}\max(a_i, a_{i+1}, \cdots, a_j) \times \gcd(i,j)^2\right) \pmod{10^9+7}$$

where gcd is the greatest common divisor function.

Input

The first line of input contains an integer N $\left(1 \le N \le 5 \times 10^5\right)$, indicating the size of the sequence.

The second line contains N positive integers, a_i $\left(1 \le a_i \le 10^9\right)$, separated by spaces, representing the sequence of numbers.

Output

On a single line print the answer to the problem.

Sample input 1	**Sample output 1**
`3` `1 2 3`	`44`

Notes

In the example, we have the sequence 1, 2, 3 with the following ranges:

1. $[1,1],[2,2],[3,3]$ whose maxima are $\max(a[1,1]) = 1$, $\max(a[2,2]) = 2$, $\max(a[3,3]) = 3$
2. $[1,2],[2,3]$ whose maxima are $\max(a[1,2]) = 2$, $\max(a[2,3]) = 3$
3. $[1,3]$ whose maximum is $\max(a[1,3]) = 3$

Thus, the sum is developed as

$$\begin{aligned}&\max(a[1,1]) \cdot \gcd(1,1)^2 + \max(a[2,2]) \cdot \gcd(2,2)^2 + \\ &\max(a[3,3]) \cdot \gcd(3,3)^2 + \max(a[1,2]) \cdot \gcd(1,2)^2 + \\ &\max(a[2,3]) \cdot \gcd(2,3)^2 + \max(a[1,3]) \cdot \gcd(1,3)^2\end{aligned}$$

Replacing the values, we get

$$\begin{aligned}&1 \cdot \gcd(1,1)^2 + 2 \cdot \gcd(2,2)^2 + 3 \cdot \gcd(3,3)^2 + \\ &2 \cdot \gcd(1,2)^2 + 3 \cdot \gcd(2,3)^2 + 3 \cdot \gcd(1,3)^2.\end{aligned}$$

and then

$$\begin{aligned}&1 \cdot 1^2 + 2 \cdot 2^2 + 3 \cdot 3^2 + 2 \cdot 1^2 + 3 \cdot 1^2 + \\ &3 \cdot 1^2 = 1 + 8 + 27 + 2 + 3 + 3 = 44.\end{aligned}$$

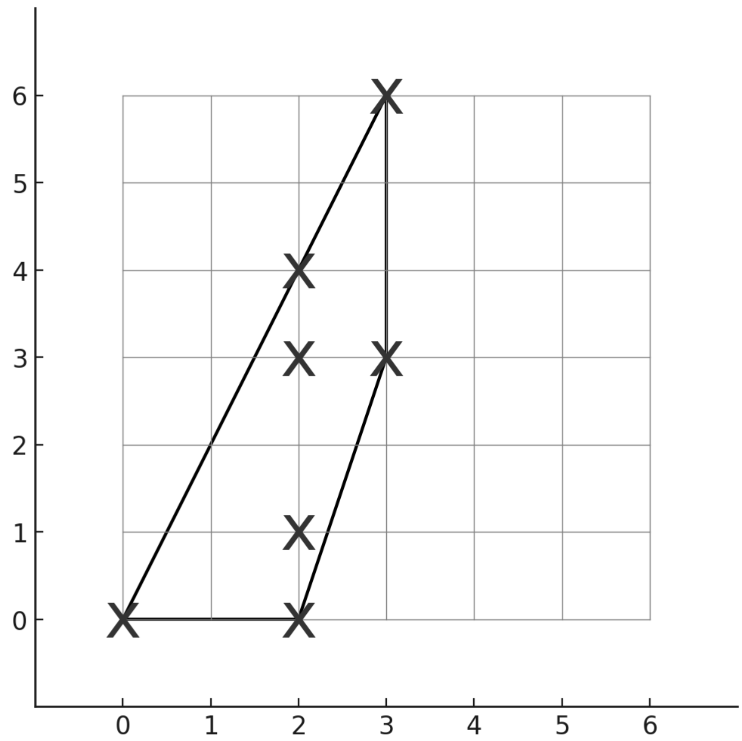
Finally, taking this result modulo $10^9 + 7$, we still get 44 since $44 < 10^9 + 7$.

Problem B – Birthday Cake

Isabel's birthday party is about to end, and like at almost any party, it's time to eat cake. Isabel prepared a cake for her party in the shape of a convex polygon, which still needs candles. Isabel likes things to look orderly, so she only puts candles in positions on the cake that have integer coordinates. Isabel has decided to add candles only at integer coordinates on the cake that meet the following characteristics:

- The coordinate is inside the cake or on its boundary.
- The coordinate can be the endpoint of a line segment that touches at least N points with integer coordinates inside the cake or on its boundary.

Given P vertices of the polygon that form the cake, and N (the number of points that the line segment must touch), what is the maximum number of candles Isabel can place on her cake?



In the picture above the birthday cake has 4 vertices $(0, 0)$, $(2, 0)$, $(3, 3)$, and $(3, 6)$, if $N = 4$, then 7 candles can be put on the cake. The coordinates where the candles are put are shown with an X.

Input

The first line of input contains two integer numbers separated by a space P $(1 \leq P \leq 1000)$ and N $(2 \leq N \leq 10^4)$, representing the number of vertices of the polygon that form the cake and the minimum number of integer coordinates the segments must touch.

Each of the next P lines contains two integer numbers separated by a space x_i, y_i, $(0 \leq x_i, y_i, \leq 5000)$ representing the coordinates of the i-th vertex in the polygon.

Output

Print a line with a single integer number, the number of candles that can be put on the cake following Isabel's conditions.

Sample input 1	**Sample output 1**

`4 4`
`0 0`
`3 3`
`2 0`
`3 6`
	`7`

Sample input 2	**Sample output 2**

`5 2`
`0 0`
`1 0`
`2 0`
`3 3`
`3 6`
	`13`

Sample input 3	**Sample output 3**

`3 5`
`0 0`
`3 10`
`5 0`
	`24`

Problem C – Cuckoo Synchronization

Artem is excited to be visiting his grandma for the weekend since he is able to play around with her vast collection of N cuckoo clocks. Artem has put a lot of effort on synchronizing the clocks so that all of them emit their first cuckoo at minute 1. But having such a variety of clocks comes with a problem: each clock will repeat their sounds at different intervals.

To be more specific, the $i-$th $(1 \leq i \leq N)$ cuckoo clock will make a sound every i minutes after their synchronized sound. For example, for $N \geq 3$ the $i = 3$ clock will keep emitting sounds every 3 minutes after it's first cuckoo at time 1. So Artem will be able to listen to it's beautiful sound on the $1, 4, 7, 10, \ldots$ minutes.

Excited for all of his hard work, Artem wants to show the experiment to his grandma. Currently it's time 0 and he knows that his grandma will be back from the store in T minutes. Artem wants you to help him figure out how many clocks will make their cuckoo sound exactly when his grandma arrives!

Input

The first line contains an integer $Q(1 \leq Q \leq 100)$ - the number of test cases.

For each test case, the only line contains two integers - $N(1 \leq N \leq 10^9)$ and $T(1 \leq T \leq 10^9)$, for the number of cuckoo clocks and the time in minutes when Artem's grandma is back, respectively.

Output

For each test case, output the number of clocks that make the cuckoo sound at exactly T minutes.

Sample input 1	**Sample output 1**
5 5 1 10 5 10 6 5 3 6 11	5 3 2 2 3

Sample input 2	**Sample output 2**
3 1000 647 1000000 123456 1000000000 1000000000	8 4 20

Problem D – Dueling Digits

In the land of Numeria, two friends, Alice and Bob, are fascinated by numbers. Recently, they discovered a curious property about certain pairs of numbers and decided to explore it further. They are interested in finding pairs of numbers with the following properties:

1. Both numbers have N digits.
2. The sum of the digits of Alice's number is equal to the sum of the digits of Bob's number.
3. For any digit position i, the i-th digit of Alice's number is different from the i-th digit of Bob's number.
4. Both numbers cannot start with the digit zero.

You have Q queries, and for each query, you need to determine how many pairs of numbers exist that satisfy these conditions for a given number length N.

Input

The first line contains an integer Q ($1 \le Q \le 800$), the number of queries.

Each of the next Q lines contains a single integer N ($1 \le N \le 800$), representing the length of the numbers.

Output

For each query, print a single integer representing the number of valid pairs of numbers that satisfy the conditions for the given length N, because this number can be very large print it modulo $10^9 + 7$.

Sample input 1	Sample output 1
1 2	480

Sample input 2	Sample output 2
2 3 4	30612 2437704

Problem E – Egotistical Command Chain

The ICPC is an organization made up of lots of competitive programmers, but it's very chaotic, so you have been tasked with assigning a command chain. A command chain can be seen as a directed graph where the vertex (i, j) indicates that the i-th competitive programmer can give orders to the j-th competitive programmer.

You know competitive programmers are very egotistical people, so they will be mad unless they have power over at least a_i people (this number can be different for each person). But if they have control over more than a_i persons, they will go mad with power, so you want to make the command chain so that every person has control over exactly a_i persons. You also don't want to have a cycle, that means, a path following the edges of the graph, such that you begin and end on the same person.

We say a person i has power over a person j if there is a sequence of people $b_1, b_2, \ldots, b_k$ such that $b_1 = i$, $b_k = j$, and b_h can give orders to b_{h+1} for all $1 \leq h < k$. Notice that a person always has power over itself.

To save resources, and so it is not that complicated, you can use at most 10^6 edges on your graph.

Input

The first line of input contains an integer N ($1 \leq N \leq 10^5$) — The number of people in the organization.

The second line of input contains N integers a_i ($1 \leq a_i \leq N$) ($a_1 + a_2 + \ldots + a_N \leq 10^6$) — The i-th integer is the number of people that the i-th programmer must have power over.

Output

If it's impossible to create the command chain with the restrictions of the problem, print `-1`.

Otherwise, print m — The number of edges in your graph. On the next m lines print two integers u_i and v_i indicating that u_i can give orders to v_i.

It can be proven that with the conditions of the problem, it is possible to construct the graph with at most 10^6 edges.

Sample input 1	**Sample output 1**

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

Sample input 2	**Sample output 2**
`5` `5 5 5 5 5`	`-1`

Problem F – Fair Prize

John is at the fair and has finally won a prize in the marble game. In the prize selection, there are n different prizes arranged in a row, each of the n prizes has a label v_i $(1 \leq i \leq N)$ that represents the value of the prize. John has scored p points in the marble game, and, according to the marble game rules, he can select a prize that has a value less than or equal to p.

Given the values of the n prizes, can you help John select the prize with the highest value that he can choose?

Input

The first line of input contains two integer numbers separated by a space, n $(1 \leq n \leq 1000)$ and p $(1 \leq p \leq 1000)$, representing the number of prizes in the marble game and the amount of points John scored in the game, respectively.

The second and last line of input contains n integer numbers separated by a space, where the i-th number represents the value v_i $(1 \leq v_i \leq 1000)$ of the i-th prize.

Output

Output a line with a single integer number, the value of the prize with the highest value that John can choose.

Sample input 1	Sample output 1
`5 10` `4 2 4 3 9`	`9`

Sample input 2	Sample output 2
`1 10` `10`	`10`

Sample input 3	Sample output 3
`3 6` `6 2 4`	`6`

Problem G – Graphoria's Villages Visit

In the mythical land of Graphoria, there exists a magical forest composed of N interconnected trees forming a single grand tree. Each node of the tree represents a village, and each village sends a traveler to every other village exactly once, using the unique paths between them.

The wise king of Graphoria, King Algor, has issued a decree. He wants to know which paths (edges) in the forest are the most traversed by the travelers and how frequently they are used. The kingdom's prosperity depends on understanding these patterns, as it will help in maintaining the paths and ensuring the smooth transit of its inhabitants.

You are given a tree with N nodes (numbered from 1 to N) and $N-1$ edges. Each node in the tree represents a village. Every village sends a traveler to visit every other village exactly once, following the unique path between them in the tree.

Your task is to determine:

1. The maximum number of times any single edge in the tree is visited.
2. The number of edges that are visited exactly this number of times.

Input

The first line contains a single integer N ($2 \leq N \leq 10^6$), the number of nodes in the tree.

Each of the next $N-1$ lines contains two integers U and V ($1 \leq U, V \leq N$) indicating there is an edge between nodes U and V.

Output

Print two integers. The first integer is the maximum number of times any single edge is visited. The second integer is the number of edges that are visited exactly this number of times.

Sample input 1	Sample output 1
7 1 2 1 3 2 4 2 5 3 6 3 7	12 2

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

Problem H – Hiring Candidates Game

There has been a selection process in your company. According to the evaluation criteria, there is a tie between n candidates. For the final selection phase, your Human Resources coordinator has designed a game to possibly filter some of them since they want to reduce the personnel cost.

The game goes as follows:

1. The n candidates form a circle, all of them are looking to its center. Then, some candidate is assigned with the id 1, the one to its left is assigned with the number 2 and so on.
2. Then, two supervisors s_1 and s_2 stand behind of the candidates with ids 1 and n, respectively.
3. If the number of candidates left is at most 2, then finish the game and hire those candidates.
4. Otherwise, the supervisor s_1 moves counting r candidates (including its own position) in clockwise direction and then stops and the supervisor s_2 moves counting c candidates (including its own position) in counter-clockwise direction and then stops. Then, there are two possibilities:
   (a) Both s_1 and s_2 reached the same candidate. In this case, remove the candidate from the circle and hire him/her.
   (b) s_1 and s_2 reached different candidates. In this case, remove both and don't hire any of them.
5. Return to Step 3.

Given the scenario of the game, compute the candidates that will be hired according to the game.

Input

The first line of input contains three integers n, r and c ($1 \leq n \leq 10^4$ and $1 \leq r, c \leq 10^5$) — The number of candidates, the number of candidates that s_1 counts while moving and the number of candidates that s_2 counts while moving, respectively.

Output

Print a single line — The sequence of ids of the hired candidates in ascending order.

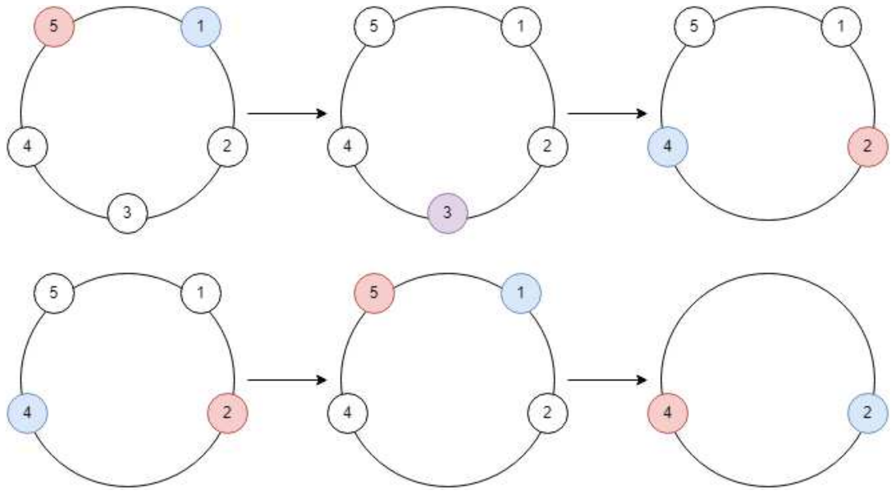
Sample input 1	Sample output 1
5 3 3	2 3 4

Sample input 2	Sample output 2
4 4 3	1 3

Sample input 3	Sample output 3
6 5 2	1 2 5 6

Notes

The following is the process of the game until it stops for the first sample case:



The candidate with id 3 is hired after the first iteration, then the candidates with ids 1 and 5 are removed and rejected after the second iteration and only the candidates with ids 2 and 4 are left. Since there are only 2 candidates, the game stops and both of them are hired. In the end, the candidates with ids 2, 3 and 4 are hired.

Problem I – Intersection of Hyperrectangles

You are given n d-dimensional hyperrectangles with sides parallel to the axes numbered from 1 to n. Each of these hyperrectangles is defined by the region of all points with real coordinates $(x_1, x_2, \ldots, x_d)$ such that $l_i \leq x_i \leq r_i$ (for $1 \leq i \leq d$). For each hyperrectangle, $2 \cdot d$ integers $l_1, r_1, l_2, r_2, \ldots, l_d, r_d$ are given.

You can do the following operation on the hyperrectangles:

- Select one hyperrectangle and move it a unit along an axis. More formally, select one hyperrectangle and a i $(1 \leq i \leq d)$ and set l_i to $l_i + 1$ and r_i to $r_i + 1$, or set l_i to $l_i - 1$ and r_i to $r_i - 1$.

Answer q queries. In each query, you are given two integers L and R. You have to find the minimum number of operations so the intersection of the hyperrectangles numbered from L to R is **non-empty**. In other words, there must exist a point that is contained inside all such hyperrectangles. A point in the boundary of the hyperrectangle is said to be inside the hyperrectangle.

Input

The first line contains two integers n and d $(1 \leq n \leq 10^4$ and $1 \leq d \leq 50)$ – the number of hyperrectangles and the number of dimensions.

The following n lines contain the description of the hyperrectangles. Each line contains $2 \cdot d$ integers $l_1, r_1, l_2, r_2, \ldots, l_d, r_d$ $(-10^9 \leq l_i < r_i \leq 10^9$ for $1 \leq i \leq d)$ – the description of the hyperrectangles.

The following line contains an integer q $(1 \leq q \leq 10^4)$ – the number of queries.

The following q lines contain the description of the queries. Each line contains 2 integers L and R $(1 \leq L \leq R \leq n)$ – the range of each query.

Output

Print q lines with the answers to the queries.

Sample input 1	Sample output 1
3 2 2 4 2 4 0 2 0 2 -2 0 -2 0 3 1 2 2 3 1 3	0 0 4

Sample input 2	Sample output 2
2 3 1 2 1 2 1 2 0 3 0 3 0 3 1 1 2	0

Sample input 3

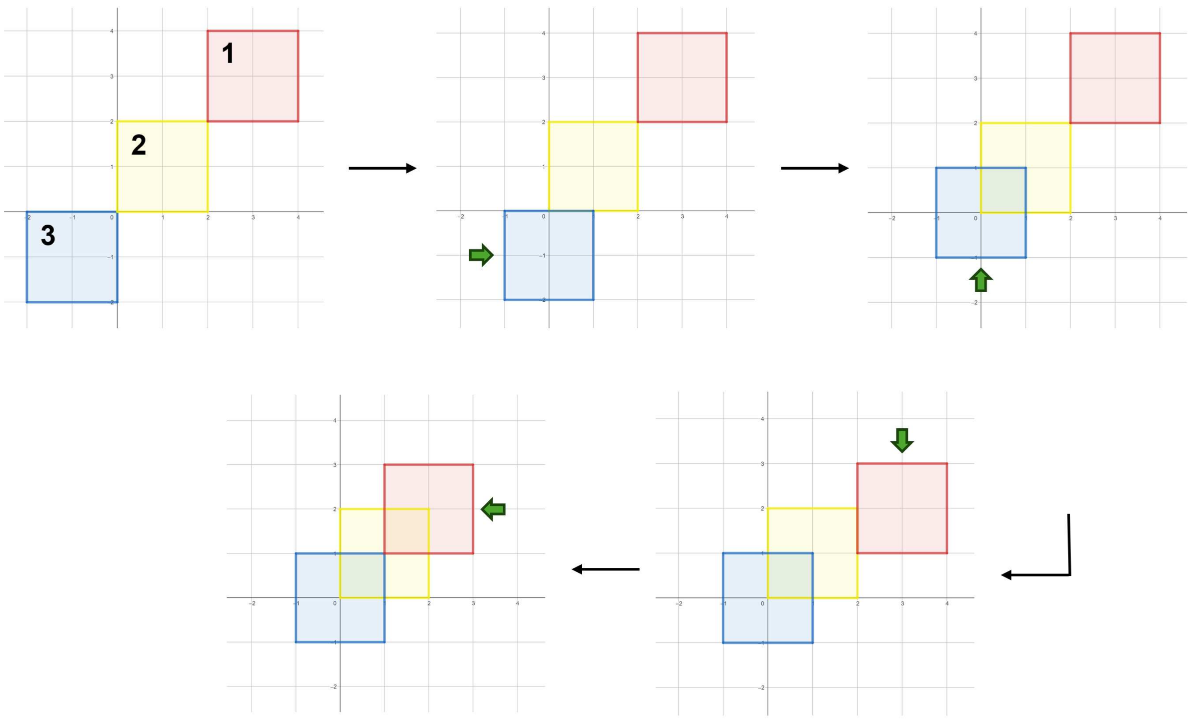
```
4 2
0 4 2 5
5 6 1 4
2 3 3 4
1 2 0 1
3
1 3
3 4
1 4
```

Sample output 3

```
2
2
5
```

Notes

The diagram below shows the minimum number of operations needed in the third query of the first sample:



The operations are as follows:

1. Move the third hyperrectangle one unit in the positive direction of the first axis.
2. Move the third hyperrectangle one unit in the positive direction of the second axis.
3. Move the first hyperrectangle one unit in the negative direction of the second axis.
4. Move the first hyperrectangle one unit in the negative direction of the first axis.

After these 4 operations, all the hyperrectangles intersect at point $(1, 1)$.
Note that the first axis is the x-axis and the second axis is the y-axis in the diagram above.

Problem J – Japanese Samurai Fight

Banshū is an old province in Japan that is known for being home of some of the best samurais of all time. Currently, N samurais live there and are preparing themselves for their annual celebration where some of the best samurais fight against each other. As you may know, samurais rely on respect as the base for their communities. This respect is mutual, meaning that if samurai a respects samurai b, then samurai b respects samurai a, too. One caveat is that this relation **is not transitive**.

For the main event of the celebration, they want to pick two of the highest respected samurais. Since they don't want to pick specific participants yet, they want to divide all samurais into two non-empty subsets, S_1 and S_2, such that **all members**, s_1 and s_2 respectively, of each subset are good candidates for the main fight. A good candidate for the fight is a samurai s who is respected by all the members of the subset S to which it belongs. No samurai should be left out of this selection process ($|S_1| + |S_2| = N$).

They're currently unsure if this is possible, so they've come to you for help. Given that there's still some time until the celebration, you've taken the task to try and make it happen. For this, you know that if you introduce two samurais while giving them their favorite sake, it's certain that they will end up respecting each other.

Having this strategy in mind, you want to know if by introducing some samurais to respect each other, you're capable of dividing the whole population in subsets S_1 and S_2 such that the fight can take place.

But you've got to hurry! Theres not much time left and you've to pick your introductions wisely as they're not unlimited.

Input

For the first line of the input you get two integers N ($1 \leq N \leq 1000$) and M ($0 \leq M \leq \frac{N(N-1)}{2}$), that represent the number of samurais in the province and the number of already known relationships between them, respectively.

For each of the next M lines you get two integers a and b ($1 \leq a_i \neq b_i \leq N$), that represent the indices of the two samurais that already respect to each other.

Output

For the first line of the output print "YES" (without quotes) if it is possible to make the fight happen, and "NO" otherwise.

If the answer is "YES", then in the next line print K ($0 \leq K \leq \frac{N(N-1)}{4}$) - the number of introductions that will take place to make it happen.

For each of the next K lines, print two integers: u and v ($1 \leq u_i \neq v_i \leq N$), the indices of the samurais that will end up respecting each other during the i-th introduction. You can't print duplicated introductions.

Sample input 1	**Sample output 1**
`1 0`	`NO`

Sample input 2	**Sample output 2**
`2 1` `1 2`	`YES` `0`

Sample input 3	**Sample output 3**

`6 4`
`1 2`
`1 3`
`2 3`
`4 5`
	`YES`
	`2`
	`4 6`
	`5 6`

Problem K – K Happy Computers

Jose is a big computer fan, he likes them so much that he plans on buying one every week! However, due to the space in his house, he can only have k computers at the same time, so whenever he buys one that won't fit, he sells the oldest one first.

Jose also likes to keep his computers happy, so he assigns them a name the computer chooses. The computers, being computers, always choose uniformly a random integer between 1 and N (because that's how Jose sets their configuration). However, a computer won't be happy if there is another computer in the house with its same name.

Jose was recently robbed and doesn't have any computers. Help him find the expected value of weeks that will pass before a computer is sad. It can be proven that the answer can be represented as a rational number $\frac{p}{q}$ with coprime p and q. You need to output $p \cdot q^{-1} \bmod 10^9 + 7$.

Input

The first line of input contains two integers N and k ($1 \leq N \leq 10^{12}$, $2 \leq k \leq 10^6$) — The range of numbers that can be used to assign the name of a computer and the maximum number of computers that Jose can have at the same time, respectively.

Output

Print a single line — The answer to the problem modulo $10^9 + 7$.

Sample input 1	Sample output 1
`2 2`	`3`

Sample input 2	Sample output 2
`5 3`	`4`

Sample input 3	Sample output 3
`1000000000000 1000000`	`798779352`

Problem L – Lost Shoes

Summer is coming! And for Miguel's family, it means a celebration is coming, but they are a pretty weird family. Not only do they hate music and celebrate summer, but they are also shoemakers, so his mother Coco gifts the whole family with a pair of shoes every year.

However, this year Miguel's dog has made a mess and the shoes got all mixed up. Everyone got a left shoe and a right shoe, but not necessarily the shoes they were supposed to get. So now they have asked for your help to get everyone their shoes.

To do this, two people can swap a shoe between them, but only if they are of the same side (they can only swap a left shoe with a left shoe and a right shoe with a right shoe). They want to know what is the minimum number of swaps they need to do so everyone ends with their shoe.

Input

The first line contains a number N ($1 \leq N \leq 10^6$) indicating the number of people in the family.

The second line contains N numbers $a_1, ..., a_N$ ($1 \leq a_i \leq N$). The i-th number indicates who is the owner of the right shoe the i-th person has.

The third line contains N numbers $b_1, ..., b_N$ ($1 \leq b_i \leq N$). The i-th number indicates who is the owner of the left shoe the i-th person has.

Output

Print a line with a number k, the minimum number of swaps they need to do so everyone has their shoes.

Sample input 1	**Sample output 1**
2 1 2 2 1	1

Sample input 2	**Sample output 2**
3 1 3 2 2 1 3	2

Sample input 3	**Sample output 3**
5 4 5 1 2 3 3 1 4 5 2	8

Problem M – Maximizing the Sauce

Soto is a big fan of movies, and as such, he likes to go to the cinema. The thing he likes most about the cinema is the sauce, and he wants to take as much as possible. The manager knows this and has made a new rule. People can only take N cups of sauce.

But even with this rule, Soto wants to get as much sauce as possible. To do so, he plans to change the form of the cups so they are able to hold the most possible sauce. A cup of sauce can be seen as an isosceles trapezium rotated by the center of the base. We then consider the smallest of the two obtained circles as the base of the cup.

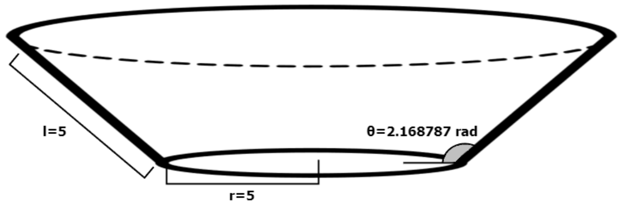


Figure 1. Answer for the first sample.

The base of cup i has a radius r_i, and sides of length l_i. Soto can choose to change the angle of the sides of the cup to be whatever he wants. Help him find the angle that maximizes the volume of sauce he can put on the cup. However, the sizes of the cups are all different, so he will ask you N questions.

Input

On the first line, you will be given an integer N ($1 \leq N \leq 10^5$), the number of cases.

On the next N lines you will be given two integers l_i and r_i ($1 \leq l_i, r_i \leq 10^4$), the length of the sides, and the radius of the base of the cup.

Output

You must print N different lines. On the i-th line you should print a real number indicating the angle in radians that maximizes the volume of the cup. Your answer will be accepted if its absolute or relative error does not exceed 10^{-6}. formally, if p is your answer, and j is the jury's answer, this should hold: $\frac{|p-j|}{max\{1,|j|\}} \leq 10^{-6}$.

Sample input 1	**Sample output 1**
3	2.16878769
5 5	1.75225402
10 53	2.42506475
1000 235	