A: Bump, Set, Spike!

Problem ID: bumpsetspike

Volleyball season is starting at your local recreational center! This means all kinds of volleyball leagues are about to be in full swing: leagues for youths, seniors, amateurs, and so on. The most prestigious and highly contested league at the rec center is the Premier Adult Coed Volleyball League. In recent years, the prestige of the league has become problematic, as all the top referees have opted to shirk their duties and compete as players, leaving only sub-par referees to oversee league matches.

To combat this, league organizers have proposed the possibility of a computer program that can play the role of referee. The games are played by typical volleyball rules: there are two teams consisting of six players each, and the program is to determine that the rallies between the two teams are valid. More specifically, for each rally, a sequential record of who touches the ball is streamed to the program, and from that it should be determined in real-time if there are any violations of the following rules:

1. A player cannot touch the ball twice in a row.
2. A team cannot touch the ball more than three times in a row.
3. If a team touches the ball twice or more before it is hit to the other team, both a man and a woman must have touched the ball at least once. A ball is considered to be "hit to the other team" once the other team makes contact with the ball, OR at the end of a rally. For example, if a rally ends with three consecutive hits from women on team `A`, that is a violation.

Other violations, such as net touches, carries, or the ball landing out of bounds will still be called by human referees, so the program need not worry about those.

The league organizers have come to you for help in creating this program. The ball is in your court!

Input

The first line contains a single integer N $(2 \leq N \leq 10^5)$, the number of touches in the rally. Then follows N lines, with the ith line describing the ith touch in the rally. Each of the N lines will contain a string consisting of three characters. The first character will be either `A` or `B`, denoting the team that touched the ball. The second character will be one of 1, 2, 3, 4, 5, or 6, denoting the specific player on the team that touched the ball. The final character will be either `W` or `M`, denoting whether the player that touched the ball was a man or a woman. Throughout a rally, the gender of players is guaranteed to remain consistent.

Output

If there was a violation in the rally, print either `A` or `B`: whichever team was first to commit a violation. If there was no violation, print `No violation`.

Sample Input 1

```
3
A1M
A1M
B5W
```

Sample Output 1

```
A
```

Sample Input 2

```
3
A1M
A2M
B2M
```

Sample Output 2

```
A
```

Sample Input 3

```
5
A4W
A5W
A1M
B3M
A2M
```

Sample Output 3

```
No violation
```

B: Bump, Set, Spike, Count!
Problem ID: bumpsetspikecount

Volleyball season is starting at your local recreational center! This means all kinds of volleyball leagues are about to be in full swing: leagues for youths, seniors, amateurs, and so on. Though the Premier Adult Coed Volleyball League is considered the most prestigious league, the Youth Girl's Volleyball League has been rising in popularity in recent years.

Again, matches in this league are played by typical volleyball rules: there are two teams consisting of six players each. Since league organizers have already solved the problem of determining if a set of touches is valid, league organizers have now set their sights on a different problem. They want to calculate the number ways to construct a valid rally consisting of exactly N touches.

Because this is not a coed league, the rules governing a valid set of touches in a rally sequence are looser, as there are no restrictions around the gender of the players. A sequence of touches is only invalid if either of the following is violated:

1. A player cannot touch the ball twice in a row.
2. A team cannot touch the ball more than three times in a row.

The league organizers have come to you for help. The ball is in your court! Since the number of ways maybe large, output the answer module $10^9 + 7$.

Input

The input consists of a single integer N $(0 \leq N \leq 5 * 10^4)$.

Output

Output the number of ways to constuct a valid sequence of touches of size N. Since the output may be large, output it modulo $10^9 + 7$.

Sample Input 1	Sample Output 1
`1`	`12`

Sample Input 2	Sample Output 2
`2`	`132`

C: Candy Consumption
Problem ID: candyconsumption

Kevin loves to eat all types of candy: chocolate, hard candy, jelly beans, gummy bears, and so on. There are N different types of candy that he is particularly enamored of at the moment, and unfortunately for his dentist, he maintains a virtually unlimited stash of each type.

Kevin considers himself something of a candy connoiseur; he is very particular about how he consumes his candy. He finds that variability in the sequence of candies that he eats is extremely important, as different qualities of the candies are really highlighted by certain sequences. For example, eating gummy bears after hard candy really accentuates the chewy texture, whereas eating jelly beans after eating dark chocolate really brings out the sweetness and flavor of the jelly beans.

Knowing this, Kevin has devised the following procedure for eating the N types of candy he currently enjoys. He selects a candy from amongst the N types at random, and eats a candy of that type. He then repeats, continuing to randomly select and consume a candy until he eats a candy type that he has eaten previously. Once he has repeated a candy type, he finds that the sequence's variability has been ruined, so he stops. Kevin calls this ordered sequence of candies that he has eaten a Candy Consumption Sequence. For example, say he randomly chooses to eat chocolate, then jelly beans, then hard candy, then gummy bears, then chocolate again. This is a Candy Consumption Sequence with 5 total candies.

Kevin realizes that there are a finite number of unique Candy Consumption Sequences. If he consumes all the candies for every possible unique Candy Consumption Sequence, how many total candies will he have eaten? Since this answer may be very large, output its remainder modulo the prime $1\,000\,000\,007$.

Input

The first line contains a single integer N ($1 \leq N \leq 10^6$).

Output

Output a single integer representing the number of candy consumed over all possible Candy Consumption Sequences, modulo $1\,000\,000\,007$.

Sample Input 1	Sample Output 1
1	2

Sample Input 2	Sample Output 2
2	16

D: Chemical Equations
Problem ID: chemicalequations

Bob is a master chemist and he has a list of N different chemical equations that detail how to create specific compounds. Each chemical equation consists of a compound and R reactants with associated quantities that can be used to make the compound. A "base compound" is defined as a reactant that cannot be further reduced into other reactants according to the list of chemical equations. Given this list of chemical equations and a compound that is in the list of chemical equations (possibly only as a reactant), output an alphabetized list of **base compounds** and quantities that can be used to form the given compound. Note that the chemical equation that forms the given compound will not necessarily contain only base compounds. You are guaranteed that the list of chemical equations will not form a cycle, meaning that there exists no compound that can be used to make its constituent reactants. Additionally, there will be at most one equation detailing how to create a compound for each compound, so there will be a single unique answer.

Input

The first line contains a single integer N such that $1 \leq N \leq 1\,000$. Next will follow N chemical equations.

Each chemical equation starts with a line that gives the name of the compound as a string followed by an integer, R, with $1 \leq R \leq 10$, indicating that R reactants are used to make the compound. The next R lines of the chemical equation each consist of an integer q_i (the quantity of the component reactant needed to make the compound), with $1 \leq q_i \leq 10$, and the name of the component reactant given as a string.

The final line of the input will contain a compound name that was previously in the list of chemical equations. You will output all the base compounds that are needed to form this compound.

Output

Given that you can form the desired compound with B base compounds, output B lines sorted alphabetically, with the ith line containing the ith base compound's name and the quantity of that base compound needed to form the given compound.

Sample Input 1

```
1
AL2O3 2
2 AL
3 O
AL2O3
```

Sample Output 1

```
AL 2
O 3
```

Sample Input 2

```
2
CO2 2
1 C
1 O2
O2 1
2 O
CO2
```

Sample Output 2

```
C 1
O 2
```

E: Con-cat-nation
Problem ID: concatnation

In Catopolis, there is a widely held belief that the neighboring nation Pupridge is preparing to invade. The coolest cats have decided to send their best concats into Pupridge as spies, but there's a problem. Pupridge has a bullet-proof identification scheme: every pet seeking entry must present an identifier in the form of a string consisting only of lower-case letters "a" to "z". The Border Collies check two things: that the identifier's length is within certain allowed bounds, and that every character in the identifier is followed by an allowed successor character.

More specifically, the Border Collies have decided on a minimum allowed identifier length x, a maximum allowed identifier length y, and, for each lower-case letter c, a list of letters that are allowed to follow c. Characters at the end of identifiers are not checked for valid successors. For example, if the only character allowed to follow "a" is "a", then "aaaaa" is a valid identifier, while "az" is not, as "z" is not allowed to follow "a".

Cats cannot type, so they cannot forge identifiers!

Of course, the crafty cats have found a workaround: they can use a scanner and printer to append identifiers of fellow cats to their own identifiers! Resources are scarce, so each cat can only append one other identifier to their own, and this identifier has to be from the pool of existing cat identifiers (note that cats are allowed to append a copy of their own identifier).

For each cat, it is your job to figure out if they can pass verification under this scheme!

Input

The first line contains space-separated integers N ($1 \leq N \leq 10^5$), x, y ($0 \leq x \leq y \leq 20$): the number of cats, the minimum allowed length of the identifier, and the maximum allowed length of the identifier. 26 lines follow, denoting the valid successors of the characters "a, b, c, …, z", respectively. Each line has the format k S, where k is the number of valid successors the character has and S is a string of length k containing all of the valid successors. Finally, N lines follow, each containing a string id_i ($1 \leq |id_i| \leq 20$), giving the identifier of the ith cat.

Output

For each cat, output "Feline good!" if the cat can pass verification and " Get meowt!" if they cannot. The output should follow the order given in the input, and each cat's output should be on a new line.

Sample Input 1

```
3 10 20
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
paws
catnip
purrfect
```

Sample Output 1

```
Feline good!
Feline good!
Feline good!
```

Sample Input 2

```
2 10 20
1 a
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
26 abcdefghijklmnopqrstuvwxyz
zzzzza
dogs
```

Sample Output 2

```
Get meowt!
Feline good!
```

F: Dominant Conversations
Problem ID: dominantconversations

Whenever people get together, there are some people who dominate regions of the conversation. We are given a record of a N-second long conversation which details who was talking at each second (people talk in second-long intervals). For each of K people, your job is to find out the number of regions of the conversation they dominate. A region is a chunk of time from second i to second j inclusive, with $i \leq j$, and it is dominated by person k if more that half of the entries in the range $[i, j]$ of the record are k.

Input

The first line contains space-separated integers n, k such that $1 \leq n, k \leq 10^6$. The second line contains n space separated integers a_i, the entries in the record. $0 \leq a_i < k$

Output

Output k lines, the ith of which denotes the number of regions dominated by the ith person (0-indexed).

Sample Input 1

```
5 1
0 0 0 0 0
```

Sample Output 1

```
15
```

Sample Input 2

```
5 2
0 1 1 1 1
```

Sample Output 2

```
1
13
```

G: Leaky Cauldron
Problem ID: leakycauldron

In a faraway magical land, there exists a very large cauldron. Unfortunately, it is leaking today! The cauldron holds a massive amount of water, but is leaking onto a rectangular stone area. The cauldron is floating right above the center square of the stone area, and over time, the leaked water will spread from the center square outward.

Different parts of the rectangular stone area vary in height. The stone area can be broken up into an N by M grid, with each square in the grid potentially having a different height. The water flows down into the center square of the grid (which is guaranteed to have the lowest height). After h seconds, the water level in the center square will be at height h. Once the water level reaches height h, any squares that have height h or below that are connected to the center square by already submerged squares will be covered with water. Thus, as the water level gradually rises, more squares become submerged. This water is magic; it will instantly expand to submerge adjacent squares with height less than the height of the water, so don't worry about the rate of flow from the cauldron.

At every second up until H seconds, find the area of the largest sub-rectangle in the N by M grid that is completely submerged by water. That is, every square of the rectangle should be covered by water for the rectangle to qualify.

Input

On the first line, there will be 3 integers N, M, and H respectively, with $1 \leq N, M < 200$ and $1 \leq H \leq 2\,500$. Then, the following N lines will each contain M space separated integers, each with a single height h, where $0 \leq h \leq H$, giving the heights of the N by M grid. It is guaranteed that N, M are odd. The height of the center square will always be 0, and it will always be the lowest in the grid.

Output

$h + 1$ lines, where the i^{th} line is the area of the largest rectangle of water when the water height has reached height i in the middle square.

Sample Input 1

```
5 5 5
5 2 4 2 5
2 3 2 3 2
3 2 0 2 3
4 2 2 2 4
5 2 3 2 5
```

Sample Output 1

```
1
1
6
12
15
25
```

H: Martian Patrol
Problem ID: martianpatrol

Being so close to a nuclear superpower, Martians take their defense very seriously. The Martian military base is structured as a binary tree rooted at 1, with nodes numbered $1 \dots N$. Some Martian soldiers patrol paths on the tree, moving from node to node.

The Martian patrol system is very complex. Every Martian patrol soldier is assigned two nodes as endpoints for their patrol and moves back and forth between these endpoints, always taking the shortest path. No two soldiers share an endpoint, and a soldier's left and right endpoints cannot be the same.

The patrol schedule works as follows. At the beginning of the day, all Martian soldiers start at their left endpoints. Each soldier starts moving from their left to their right endpoint sometime in the next $N - 1$ seconds, traversing one edge every second. This time is randomly chosen for each soldier independently by the Martian's sophisticated algorithm to prevent patrol timings becoming predictable. While there is an element of randomness, the start time for each soldier is always chosen such that the soldier will reach their right endpoint before (or as soon as) the $N - 1$ seconds ends. Once a soldier starts moving, they keep moving until they reach their right endpoint. This $N - 1$ seconds is considered a patrol cycle. This procedure then repeats, only this time soldiers start from their right endpoint and move to their left, and so on.

(For any given node n, all nodes in its left subtree are considered "to the left", and similarly all nodes in its right subtree are considered "to the right". A soldier at node i is "on the left" of a soldier at node j if and only if i is in j's left subtree, or i is in the left subtree of node i and node j's lowest common ancestor and j is in its right subtree. A soldier being " on the right" of another soldier is defined similarly.)

The soldiers get lonely patrolling the Martian base all day, so they like to gossip with their buddies. If it is *possible* for a soldier to meet (be at the same node or edge as) another soldier during their patrol given some start time assignments, then the soldiers consider each other buddies. Soldiers become sad if any of their buddies is always on one side during a patrol cycle (always on the left or at the same node, or always on the right or at the same node), as when they gossip it hurts their ear.

The Martians are also planning to add a super secret room to the military base. This room will be added as a child of some node in the base without violating the binary nature of the tree. Once this room is added, Martian soldiers will be allowed to have it as one of their patrol endpoints.

Your job is to count the number of ways to add the super secret room and assign patrol endpoints to any number of soldiers such that the assignment of endpoints is "non-sad". An arrangement is considered "non-sad" if it is impossible for any soldier to be sad, regardless of which starting times the algorithm picks. Two arrangments are considered different if and only if there exists a node in one arrangement not in the other, or a soldier's path in one arrangement doesn't exist in the other. Note that soldiers themselves are not considered distinct. The answer may be large, so output it modulo $10^9 + 7$.

Input

The first line contains an integer N ($2 \leq N \leq 10^6$). N lines follow, with the ith (1-indexed) line having two integers l_i, r_i, denoting the left and right child of node i. A 0 denotes that there is no child.

Output

A single integer, the number of non-sad arrangements of soldiers and rooms $\mod 10^9 + 7$.

Sample Input 1	Sample Output 1

3
2 3
0 0
0 0
	36

Sample Input 2

```
10
2 5
3 0
4 0
0 0
6 10
7 8
0 0
9 0
0 0
0 0
```

Sample Output 2

```
63778
```

I: Pigeon X
Problem ID: pigeonx

M bad pigeons in a mansion
Wrist on Milly Rock them diamonds on me dancin'
When you workin' hard then your money start expandin'
I got model pigeon wanna peck me like some candy
And them seeds come in handy
Last name Savage pigeon, no I'm not Randy

21 Savage has M bad pigeons in a mansion with N rooms. Each pigeon is in a different room. He also has M seed stashes stored in rooms across the mansion. The mansion can be represented as a weighted undirected graph where each room is a node and each corridor is an edge between two nodes. 21 Savage desires to pair pigeons with a seed stash so that the total distance each pigeon has to travel to reach its seed is minimized.

Help 21 Savage match pigeons with seed!

Input

The first line of input has two space separated integers N and M, the number of rooms and the number of pigeons and seed stashes. It is guaranteed that $1 \leq N \leq 500$ and $1 \leq 2M \leq N$.

N lines follow. Each line has N space separated integers. The jth entry on the ith line represents the length of the corridor from room i to room j. It is guaranteed that the length of the corridor from room i to room j is the same as the length of the corridor from room j to room i. It is also guaranteed that the length of the corridor from room i to room i for all i is 0. Finally the lengths $l_{i,j}$ of the corridor between room i and room j satisfies $0 \leq l_{i,j} \leq 10\,000$ for all i and j.

The next line has M space separated unique integers representing the room locations of the M pigeons. It is guaranteed that all integers F_i satisfy $0 \leq P_i < N$.

The final line has M space separated unique integers representing the room locations of the M seed stashes. It is guaranteed that all integers F_i satisfy $0 \leq F_i < N$.

Output

Output a single integer, the minimum total distance that all pigeons have to travel to eat a unique seed stash.

Sample Input 1

```
4 2
0 2 2 2
2 0 2 2
2 2 0 2
2 2 2 0
0 1
2 3
```

Sample Output 1

```
4
```

J: Red Light, Green Light
Problem ID: redlightgreenlight

A recent study has shown that long commutes can have negative effects on one's happiness. Upon reading this study, Jay had suddenly become quite unhappy with his long commute to work; it seems the study is true after all. Jay now has frequent dreams in which he has a short daily commute, and these pleasant dreams often cause Jay to sleep through his alarms in the morning.

On the day of his Very Important Meeting, Jay once again oversleeps, to his dismay. As he leaves his house at time 0, he realizes he only has T minutes to reach his workplace. The city Jay works in contains N intersections (labeled 1 to N) and M one-way roads between intersections that vary in length. His house is located at intersection 1, and his workplace is located at intersection N.

The real culprits responsible for Jay's long commute are the traffic lights that are located on every road. Jay has driven around the city so much that he's realized that each light operates on a cycle: for the ith road, the light is green for g_i minutes, then red for r_i minutes, then green for g_i minutes, and so on. Jay has also figured out the specific values of g_i and r_i for every road in the city, and he has calculated the first time t_i after time 0 (when Jay leaves his house) that each road's traffic light is green. Here, t_i will fall somewhere in the inclusive range $[0, r_i]$. Jay can only take a road if the light is green; otherwise, he must wait at the intersection or choose to take a different road. Once he enters a road, he can continue to the end even if the light turns red before he reaches the next intersection. If Jay arrives at an intersection right as an outgoing road's light switches, he must obey the new color of the light.

While Jay wants to make it to his meeting on time, he also wants to avoid driving too fast. Knowing all this information about the city's traffic lights, Jay wants to know the minimum speed he needs to drive in order to make it to his meeting in time.

Input

The first line of input contains three space-separated integers, N ($2 \leq N \leq 50\,000$), M ($1 \leq M \leq 50\,000$), and T ($1 \leq T \leq 10^6$). Then follows M lines, the ith of which describes the ith road in the city. Each of the M lines will contain six space-separated integers, u_i, v_i ($1 \leq u_i, v_i \leq N$), l_i ($1 \leq l_i \leq 10^6$), g_i, r_i ($1 \leq g_i, r_i \leq 10^4$), t_i ($0 \leq t_i \leq r_i$), denoting that the ith road is from intersection u_i to intersection v_i, has a length of l_i feet, and the period of the road is described by g_i, r_i, and t_i (as explained in the statement above). There will be at most one road from city u to city v for all u, v, and there will be no roads from a city to itself (note that there may be a road from u to v AND v to u).

It is guaranteed that Jay will be able to reach his workplace from home.

Output

Output the minimum speed Jay needs to drive in feet per minute in order to reach his Very Important Meeting on time. Answers within a relative or absolute error of 10^{-6} will be accepted.

Sample Input 1

```
4 4 12
1 2 4 1 1 0
1 3 6 2 2 1
2 4 8 3 4 2
3 4 4 4 6 3
```

Sample Output 1

```
1.000000
```

K: Undetermined Determinant
Problem ID: undetermineddeterminant

Two of the classes freshmen at the University of Texas often take are linear algebra and discrete math. The former teaches vectors, matrices, and operations on them, while the latter covers a variety of topics including combinatorics and graph theory.

John is a freshman at UT, and it is finals season. His professors gave study guides for their final exams, and one of the bullet points in the linear algebra study guide is the following:

- Given a matrix, you should be able to compute its determinant.

Here's a bullet point in the discrete math study guide:

- Given a graph, you should be able to write down its adjacency matrix.

In the course of his studying, John realized that these problems can be combined naturally: given a graph, compute the determinant of its adjacency matrix. After solving this problem a few times, John began to grow bored, and he wondered if he could find some information about the answer to this question for a whole class of graphs. He wrote the following question on a blank page in his notebook before going to sleep:

- Given a positive integer N, consider a uniform random labeled tree on N vertices. What is the expected value of the determinant of its adjacency matrix?

Now John is asleep and the problem is haunting his dreams. Can you help him solve it?

It can be shown that the answer is a rational number that can be written in the form p/q, with $q \not\equiv 0 \pmod{998\,244\,353}$. You should output your answer as an integer in $[0, 998\,244\,353)$ equivalent to pq^{-1} modulo $998\,244\,353$.

Definitions

A tree on N vertices is a connected undirected graph with N vertices and $N-1$ edges. A labeled tree is one whose vertices are labeled with the integers $1, \ldots, N$. Two labeled trees are distinct if they do not have the same set of edges, where an edge is defined as an unordered pair of integers in $\{1, \ldots, N\}$. Note that two distinct labeled trees may have the same structure, but different vertex labelings.

Input

The input consists of a single line containing a single integer, N ($1 \leq N \leq 10^6$), the number of vertices in the uniform random labeled tree.

Output

Output a single integer in $[0, 998\,244\,353)$, the answer to the problem as a rational number modulo $998\,244\,353$.

Sample Input 1	Sample Output 1
`1`	`0`

Sample Input 2	Sample Output 2
`2`	`998244352`

L: Uzbek Trek
Problem ID: uzbektrek

Traveler Todd plans on making the trek from Kazakhstan to Turkmenistan. In order to do so, he must navigate through the great nation of Uzbekistan.

Unfortunately, Uzbekistan imposes a tax on all foreign travelers who pass through it. Uzbekistan has imposed a particularly peculiar tax on Todd. While trekking through Uzbekistan, Todd must wear special shoes with the blue, white and green flag of Uzbekistan stiched onto them. Each shoe only lasts for a certain amount of footsteps. Specifically, the kth type of shoe lasts for exactly p^k footsteps, where $k \geq 0$ and p is some predetermined value.

Being the prepared traveler he is, Todd calculates that it will take him n footsteps to travel through Uzbekistan. He then goes to the Uzbek Shoe Depot where he plans to purchases all the shoes he needs to complete his journey. The Uzbek Shoe Depot is a luxury store, so Todd can order as many shoes as he likes. For each shoe that Todd orders, he specifies the type k of shoe that he wants. Todd would hate to waste the durability of his purchased shoes; help him determine how many ways he can order sets of shoes so that he can wear all the shoes in sequence to travel **exactly** n footsteps! The answer may be large, so output its remainder modulo $10^9 + 7$.

Note that two sets of shoes are considered the same if they have the same number of shoes of each type. It does NOT matter what order Todd wears the shoes in.

Input

The first line contains space-separated integers n and p such that $1 \leq n \leq 10^9$ and $10^2 \leq p \leq 10^9$.

Output

Output a single integer, the number of unique sets of shoes Todd can order to travel exactly n steps. Since the answer can be large, output it modulo $10^9 + 7$.

Sample Input 1	Sample Output 1
`100 100`	`2`