Nordic Collegiate Programming Contest

NCPC 2018

October 6, 2018

Problems

A Altruistic Amphibians
B Baby Bites
C Code Cleanups
D Delivery Delays
E Explosion Exploit
F Firing the Phaser
G Game Scheduling
H House Lawn
I Intergalactic Bidding
J Jumbled String
K King's Colors

Do not open before the contest has started.

Advice, hints, and general information

- The problems are not sorted by difficulty.
- Your solution programs must read input from *standard input* (e.g. `System.in` in Java or `cin` in C++) and write output to *standard output* (e.g. `System.out` in Java or `cout` in C++). For further details and examples, please refer to the documentation in the help pages for your favorite language on Kattis.
- For information about which compiler flags and versions are used, please refer to the documentation in the help pages for your favorite language on Kattis.
- Your submissions will be run multiple times, on several different inputs. If your submission is incorrect, the error message you get will be the error exhibited on the first input on which you failed. E.g., if your instance is prone to crash but also incorrect, your submission may be judged as either "wrong answer" or "run time error", depending on which is discovered first. The inputs for a problem will always be tested in the same order.
- If you think some problem is ambiguous or underspecified, you may ask the judges for a clarification request through the Kattis system. The most likely response is "No comment, read problem statement", indicating that the answer can be deduced by carefully reading the problem statement or by checking the sample test cases given in the problem, or that the answer to the question is simply irrelevant to solving the problem.
- In general we are lenient with small formatting errors in the output, in particular whitespace errors within reason, and upper/lower case errors are often (but not always) ignored. But not printing any spaces at all (e.g. missing the space in the string "`1 2`" so that it becomes "`12`") is typically not accepted.

  The safest way to get accepted is to follow the output format exactly.
- For problems with floating point output, we only require that your output is correct up to some error tolerance.

  For example, if the problem requires the output to be within either absolute or relative error of 10^{-4}, this means that

  - If the correct answer is 0.05, any answer between 0.0499 and $.0501$ will be accepted.
  - If the correct answer is 500, any answer between 499.95 and 500.05 will be accepted.

  Any reasonable format for floating point numbers is acceptable. For instance, "17.000000", "0.17e2", and "17" are all acceptable ways of formatting the number 17. For the definition of reasonable, please use your common sense.

Problem A
Altruistic Amphibians
Problem ID: altruisticamphibians
Time limit: 3 seconds

Picture from szftlgs.com, CC0

A set of frogs have accidentally fallen to the bottom of a large pit. Their only means of escaping the pit is to jump out of it. Each frog i is described by three parameters (l_i, w_i, h_i) where l_i is its leap capacity, w_i its weight, and h_i its height. The leap capacity specifies how high that frog can jump. If a frog's leap capacity is strictly larger than the depth of the pit, the frog can directly escape the pit. However, these frogs are altruistic. Rather than selfishly saving themselves and leaving the frogs with too limited leap capacity behind, they collectively aim to save as many of them from the pit as possible.

The frogs realize that if a frog A climbs up on the back of frog B before it jumps, the first frog A stands a better chance of escaping the pit: it can escape if $h_B + l_A$ is strictly larger than the depth of the pit.

Furthermore, if frog B carrying frog A on its back climbs up on the back of frog C, the situation is even better for frog A: it can now escape the pit if $h_C + h_B + l_A$ is strictly larger than the depth of the pit.

The frogs can build even higher piles of frogs this way, the only restriction is that no frog may carry other frogs of weight in total amounting to its own weight or heavier. Once a pile has been used to allow a frog to escape, the frogs in the pile jump back to the bottom of the pit and they can then form a new pile (possibly consisting of a different set of frogs). The question is simply how many frogs can escape the pit assuming they collaborate to maximize this number?

Input

The first line of input contains two integers n and d ($1 \leq n \leq 100\,000$, $1 \leq d \leq 10^8$), where n is the number of frogs and d is the depth of the pit in µm. Then follow n lines each containing three integers l, w, h ($1 \leq l, w, h \leq 10^8$), representing a frog with leap capacity l µm, weight w µg, and height h µm. The sum of all frogs' weights is at most 10^8 µg.

Output

Output the maximum number of frogs that can escape the pit.

Sample Input 1	Sample Output 1
3 19 15 5 3 12 4 4 20 10 5	3

Sample Input 2	Sample Output 2
3 19 14 5 3 12 4 4 20 10 5	2

Problem B
Baby Bites
Problem ID: babybites
Time limit: 1 second

Arild just turned 1 year old, and is currently learning how to count. His favorite thing to count is how many mouthfuls he has in a meal: every time he gets a bite, he will count it by saying the number out loud.

Unfortunately, talking while having a mouthful sometimes causes Arild to mumble incomprehensibly, making it hard to know how far he has counted. Sometimes you even suspect he loses his count! You decide to write a program to determine whether Arild's counting makes sense or not.

Input

The first line of input contains an integer n ($1 \leq n \leq 1\,000$), the number of bites Arild receives. Then second line contains n space-separated words spoken by Arild, the i'th of which is either a non-negative integer a_i ($0 \leq a_i \leq 10\,000$) or the string "`mumble`".

Output

If Arild's counting might make sense, print the string "`makes sense`". Otherwise, print the string "`something is fishy`".

Sample Input 1

```
5
1 2 3 mumble 5
```

Sample Output 1

```
makes sense
```

Sample Input 2

```
8
1 2 3 mumble mumble 7 mumble 8
```

Sample Output 2

```
something is fishy
```

Sample Input 3

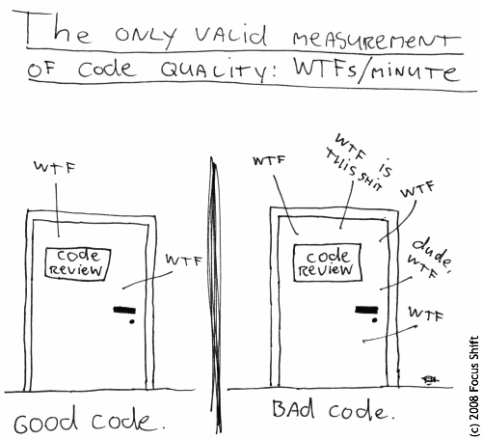
```
3
mumble mumble mumble
```

Sample Output 3

```
makes sense
```

This page is intentionally left (almost) blank.

Problem C
Code Cleanups
Problem ID: codecleanups
Time limit: 1 second



Picture by Thom Holwerda via OSnews

The management of the software company JunkCode has recently found, much to their surprise and disappointment, that productivity has gone down since they implemented their enhanced set of coding guidelines. The idea was that all developers should make sure that every code change they push to the master branch of their software repository strictly follows the coding guidelines. After all, one of the developers, Perikles, has been doing this since long before these regulations became effective so how hard could it be?

Rather than investing a lot of time figuring out why this degradation in productivity occurred, the line manager suggests that they loosen their requirement: developers can push code that weakly violates the guidelines as long as they run cleanup phases on the code from time to time to make sure the repository is tidy.

She suggests a metric where the "dirtiness" of a developer's code is the sum of the pushes that violate the guidelines – so-called *dirty* pushes – made by that developer, each weighted by the number of days since it was pushed. The number of days since a dirty push is a step function that increases by one each midnight following the push. Hence, if a developer has made dirty pushes on days 1, 2, and 5, the dirtiness on day 6 is $5 + 4 + 1 = 10$. She suggests that a cleanup phase, completely fixing all violations of the coding guidelines, must be completed before the dirtiness reaches 20. One of the developers, Petra, senses that this rule must be obeyed not only because it is a company policy. Breaking it will also result in awkward meetings with a lot of concerned managers who all want to know why she cannot be more like Perikles? Still, she wants to run the cleanup phase as seldomly as possible, and always postpones it until it is absolutely necessary. A cleanup phase is always run at the end of the day and fixes every dirty push done up to and including that day. Since all developers are shuffled to new projects at the start of each year, no dirtiness should be left after midnight at the end of new year's eve.

Input

The first line of input contains an integer n ($1 \leq n \leq 365$), the number of dirty pushes made by Petra during a year. The second line contains n integers $d_1, d_2, \ldots, d_n$ ($1 \leq d_i \leq 365$ for each $1 \leq i \leq n$) giving the days when Petra made dirty pushes. You can assume that $d_i < d_j$ for $i < j$.

Output

Output the total number of cleanup phases needed for Petra to keep the dirtiness strictly below 20 at all times.

Sample Input 1	Sample Output 1
5 1 45 65 84 346	4

Sample Input 2	Sample Output 2
3 310 330 350	3

Problem D
Delivery Delays
Problem ID: deliverydelays
Time limit: 5 seconds

Hannah recently discovered her passion for baking pizzas, and decided to open a pizzeria in downtown Stockholm. She did this with the help of her sister, Holly, who was tasked with delivering the pizzas. Their pizzeria is an instant hit with the locals, but, sadly, the pizzeria keeps losing money. Hannah blames the guarantee they put forth when they advertised the pizzeria:

> Do you have a craving for a delicious pizza? Do you want one right now? Order at Hannah's pizzeria and we will deliver the pizza to your door. If more than 20 minutes elapse from the time you place your order until you receive your Hannah's pizza, the pizza will be *free of charge*!

Picture by Clker-Free-Vector-Images via Pixabay, CC0

Even though Holly's delivery car can hold an arbitrary number of pizzas, she has not been able to keep up with the large number of orders placed, meaning they have had to give away a number of pizzas due to late deliveries.

Trying to figure out the best way to fix the situation, Hannah has now asked you to help her do some analysis of yesterday's orders. In particular, if Holly would have known the set of orders beforehand and used an optimal delivery strategy, what is the longest a customer would have had to wait from the time they placed their order until they received their pizza?

Hannah provides you with a map of the roads and road intersections of Stockholm. She also gives you the list of yesterday's orders: order i was placed at time s_i from road intersection u_i, and the pizza for this order was out of the oven and ready to be picked up for delivery at time t_i. Hannah is very strict about following the "first come, first served" principle: if order i was placed before order j (i.e. $s_i < s_j$), then the pizza for order i will be out of the oven before the pizza for order j (i.e. $t_i < t_j$), and the pizza for order i must be delivered before the pizza for order j.

Input

The first line of input contains two integers n and m ($2 \leq n \leq 1\,000$, $1 \leq m \leq 5\,000$), where n is the number of road intersections in Stockholm and m is the number of roads. Then follow m lines, the i'th of which contains three integers u_i, v_i and d_i denoting that there is a bidirectional road between intersections u_i and v_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$), and it takes Holly's delivery car d_i time units to cross this road in either direction ($0 \leq d_i \leq 10^8$). There is at most one road between any pair of intersections.

Then follows a line containing an integer k, the number of orders ($1 \leq k \leq 1\,000$). Then follow k lines, the i'th of which contains three integers s_i, u_i, t_i denoting that an order was made at time s_i from road intersection u_i ($2 \leq u_i \leq n$), and that the order is ready for delivery at time t_i ($0 \leq s_i \leq t_i \leq 10^8$). The orders are given in increasing order of when they were placed, i.e. $s_i < s_j$ and $t_i < t_j$ for all $1 \leq i < j \leq k$.

Hannah's pizzeria is located at road intersection 1, and Holly and her delivery car are stationed at the pizzeria at time 0. It is possible to reach any road intersection from the pizzeria.

Output

Output a single integer denoting the longest time a customer has to wait from the time they place their order until the order is delivered, assuming that Holly uses a delivery schedule minimizing this value.

Sample Input 1

```
4 4
1 2 2
2 3 4
3 4 1
4 1 2
3
1 4 2
3 3 3
4 3 6
```

Sample Output 1

```
6
```

Sample Input 2

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

Sample Output 2

```
8
```

Problem E
Explosion Exploit
Problem ID: explosion
Time limit: 3 seconds

In a two player card game, you have n minions on the board and the opponent has m minions. Each minion has a health between 1 and 6.

Picture by OpenClipart-Vectors on Pixabay, CC0

You are contemplating your next move. You want to play an "Explosion" spell which deals d units of damage randomly distributed across all minions. The damage is dealt one unit at a time to some remaining minion on the board. Each living minion (including your own) has the same chance of receiving each unit of damage. When a minion receives a unit of damage, its health is decreased by one. As soon as the health of a minion reaches zero, it is immediately removed from the board, before the next damage is dealt. If there are no minions left on the board, any excess damage caused by the spell is ignored.

Given the current health of all minions, what is the probability that the Explosion will remove all of the opponent's minions? Note that it does not matter if all your own minions die in the process as well, and the damage continues to be dealt even if all your own minions are gone.

Input

The first line of input contains the three integers n, m, and d ($1 \leq n, m \leq 5$, $1 \leq d \leq 100$). Then follows a line containing n integers, the current health of all your minions. Finally, the third line contains m integers, the current health of all the opponent's minions. All healths are between 1 and 6 (inclusive).

Output

Output the probability that the Explosion removes all the opponent's minions, accurate up to an absolute error of 10^{-6}.

Sample Input 1	Sample Output 1
1 2 2 2 1 1	0.3333333333

Sample Input 2	Sample Output 2
2 3 12 3 2 4 2 3	0.1377380946

This page is intentionally left (almost) blank.

Problem F
Firing the Phaser
Problem ID: firingphaser
Time limit: 3 seconds

As captain of your space ship you have never encountered a more fierce enemy than the one you have snuck upon now. You immediately bring out the big phaser cannon hoping to take out the flagship before they discover you. There is no room for mistakes and the shot will have to be perfect if you are to stand any chance at all against the flagship of the enemy.

You start charging the phaser beam and retrieve the room layout of the flagship from the archives. You are situated directly above the enemy, from where the layout of the flagship can be modeled by a two-dimensional map of the rooms of the flagship. In this map, each room is a rectangle with sides parallel to the x and y axes (rectilinear), and no two rooms intersect (not even in a single point).

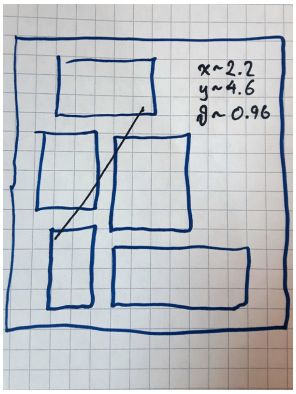


Illustration of a solution to Sample Input 1

The phaser beam is configured by giving a point (x, y) and an angle ϑ. The phaser beam will start at (x, y) and travel a distance ℓ in the direction specified by ϑ, causing severe damage to every room touched by the phaser beam. Due to this, you aim at hitting as many rooms as possible.

The phaser beam is almost fully charged and the only missing piece is an optimal configuration of the weapon. Unfortunately, it turns out to be harder than you expected. However, there are still ten seconds before the charging is completed and hence you decide to make a computer program to solve the problem.

Input

The first line of input consists of two integers r and ℓ ($1 \leq r \leq 15$, $1 \leq \ell \leq 1\,000$) where r is the number of rooms in the flagship and ℓ is the length of a shot of the phaser.

Then follow r lines, each of which contains four integers x_1, y_1, x_2, y_2 ($0 \leq x_1 < x_2 \leq 1\,000$, $0 \leq y_1 < y_2 \leq 1\,000$), indicating that there is a room in the flagship with lower left corner (x_1, y_1) and upper right corner (x_2, y_2).

Output

Output one line with the maximum number of rooms that can be hit by one phaser beam. Recall that if the beam touches a room it is counted as a hit.

You may assume that the answer is numerically stable in the following sense: if all rooms are expanded by a distance of 10^{-6} in all four directions, the answer does not change.

Sample Input 1	Sample Output 1
5 8 2 1 4 5 5 1 12 4 5 5 9 10 1 6 4 10 2 11 7 14	4

Sample Input 2	Sample Output 2
3 6 2 2 3 3 5 3 6 4 6 6 7 7	3

Problem G
Game Scheduling
Problem ID: gamescheduling
Time limit: 3 seconds

In a tournament with m teams, each team consisting of n players, construct a playing schedule so that each player is paired up against all players in all teams except their own. That is, each player should play $(m-1) \cdot n$ games.

The playing schedule should be divided into rounds. A player can play at most one game per round. If a player does not play a game in a round, that player is said to have a bye in that round.

Your task is to write a program that constructs a playing schedule so that no player has a bye in more than 1 round. In other words, the total number of rounds in the playing schedule should be no more than $(m-1) \cdot n + 1$.

Picture by Pat Paker via Wikimedia Commons, CC BY

The order of the rounds and games, and who is home and away in a game, does not matter.

Input

The input consists of a single line with two integers n and m ($1 \leq n \leq 25$, $2 \leq m \leq 25$, $n \cdot m \leq 100$), the number of players in a team and the total number of teams, respectively.

Output

Output one line per round in the playing schedule. Each line should contain a space separated list of games. A game is in the format "`<player>-<player>`". The players in the first team are denoted as `A1`, `A2`, ..., `An`; the second team `B1`, `B2`, ... `Bn` and so on.

Sample Input 1

```
3 2
```

Sample Output 1

```
A1-B2 B1-A2 A3-B3
A2-B3 B2-A3 A1-B1
A3-B1 B3-A1 A2-B2
```

Sample Input 2

```
2 3
```

Sample Output 2

```
A1-B1 A2-C2 B2-C1
A1-C1 A2-B1 B2-C2
A1-B2 A2-C1 B1-C2
A1-C2 A2-B2 B1-C1
```

Sample Input 3

```
1 5
```

Sample Output 3

```
B1-E1 C1-D1
C1-A1 D1-E1
D1-B1 E1-A1
E1-C1 A1-B1
A1-D1 B1-C1
```

This page is intentionally left (almost) blank.

Problem H
House Lawn
Problem ID: houselawn
Time limit: 1 second

Picture by David Hawgood, CC BY-SA

You have just bought a new house, and it has a huge, beautiful lawn. A lawn that needs cutting. Several times. Every week. The whole summer.

After pushing the lawnmower around the lawn during the hottest Saturday afternoon in history, you decided that there must be a better way. And then you saw the ads for the new robotic lawnmovers. But which one should you buy? They all have different cutting speeds, cutting times and recharge times, not to mention different prices!

According to the advertisement, a robotic lawnmover will spend all its time either cutting the lawn or recharging its battery. Starting from a full battery, it will cut the lawn at a given rate of c square meters per minute for a cutting time of t minutes, after which it has run out of battery. Once out of battery, it will immediately start recharging. After recharging for r minutes the battery is full again and it immediately starts cutting.

You decide that in order for your lawn to look sufficiently prim and proper, the lawnmower that you buy must be powerful enough to cut your whole lawn at least once a week *on average*. Formally, if we start the mower fully charged at the beginning of the week and run it for exactly T weeks, it needs to cut the whole lawn at least T times, for all positive integers T. But apart from this, you have no specific requirements, so among the ones that satisfy this requirement, you will simply go for the cheapest option. For the purposes of cutting your lawn, you may make the simplifying assumption that a week is always exactly $10\,080$ minutes long.

Input

The first line of input contains two integers ℓ and m ($1 \leq \ell \leq 10^6$, $1 \leq m \leq 100$), the size of your lawn in square meters, and the number of lawnmowers to consider, respectively.

Then follow m lines, each containing a string n and 4 integers p, c, t, and r, separated by commas, describing a lawnmower as follows:

- n is the name of the lawnmower, a string of at most 60 printable characters (ASCII 32 to 126) excluding '`,`', neither starting nor ending with a space,
- $1 \leq p \leq 100\,000$ is the price of the lawnmover,
- $1 \leq c \leq 100$ is the cutting rate in square meters per minute,
- $1 \leq t \leq 10\,080$ is the cutting time in minutes, and
- $1 \leq r \leq 10\,080$ is the recharge time in minutes.

Output

Output the name of the cheapest lawnmower capable of cutting your whole yard at least once a week on average. If several lawnmovers share the same lowest price, output all of their names, in the same order they were given in the input. If there is no such mower, output "`no such mower`".

Sample Input 1	Sample Output 1
7000 4 Grass Slayer 2000,9999,10,120,120 Slow-Mowe,999,1,120,240 Eco-cut X2,5499,2,25,35 Mowepower,5499,3,25,35	Eco-cut X2 Mowepower

Sample Input 2	Sample Output 2
100000 4 Grass Slayer 2000,9999,10,120,120 Slow-Mowe,999,1,120,240 Eco-cut X2,5499,2,25,35 Mowepower,5499,3,25,35	no such mower

Problem I
Intergalactic Bidding
Problem ID: intergalacticbidding
Time limit: 2 seconds

Picture by Activedia via Pixabay, CC0

Today the Intergalactic Council of Pebble Coins (ICPC) conducted an intergalactic auction of the Neutronium Chaos Pebble Coin (NCPC). This coin, which was forged in the Ancient Coin Machine (ACM), is rumored to be the key to ruling the universe.

Due to the extremely competitive nature of the auction, as well as the odd mechanics of the intergalactic currency used (far too advanced for mere mortals to understand), the auction was conducted with the following rules:

1. only one participant was allowed to make a bid at a time,
2. each participant was only allowed to make one bid, and
3. a participant making a bid had to bid at least twice the amount of the highest bid at the time.

The first participant making a bid was allowed to make a bid of any positive amount.

After the auction there were a lot of sore losers – understandably, having just lost their chance at world domination. To make the losers feel a little better and prevent possible rioting, the ICPC has decided to hold a lottery for the participants. The winners of the lottery are determined as follows. The ICPC picks a random number s. A group of participants is called *winning* if the sum of their bets from the auction is equal to s. A participant wins the lottery and receives a prize – a shiny Pebble Coin – if they belong to any winning group of participants.

Given the names of the participants, the bets that they made, and the random number s chosen by the ICPC, help them determine which participants won the lottery.

Input

The first line of input contains two integers n and s, where $1 \leq n \leq 1\,000$ is the number of participants, and $1 \leq s < 10^{1\,000}$ is the random number chosen by the ICPC.

Then follow n lines describing the participants. Each line contains a string t and an integer b, where t is the name of a participant, and $1 \leq b < 10^{1\,000}$ is the amount of his bet. The name of each participant is unique and consists of between 1 and 20 letters from the English alphabet.

Output

Output an integer k denoting the number of participants that won the lottery. Then output k lines containing the names of the participants that won the lottery, one per line, in any order.

Sample Input 1

```
5 63
Vader 3
Voldemort 7
BorgQueen 20
Terminator 40
Megatron 101
```

Sample Output 1

```
3
BorgQueen
Terminator
Vader
```

Sample Input 2

```
4 1112
Blorg 10
Glorg 1000
Klorg 1
Zlorg 100
```

Sample Output 2

```
0
```

Problem J
Jumbled String
Problem ID: jumbledstring
Time limit: 1 second

Recall that a *subsequence* of a string is any string obtained by removing some subset of characters from the string, for instance "string", "sing", "i" and "sg" are all subsequences of "string". If the same subsequence can be obtained in exactly t different ways, by removing different subsets of characters, we say that the subsequence occurs t times.

Jingfei wants to create a nonempty bit string that has the following properties:

Picture by snd63 via pixabay, CC0

1. the subsequence `00` occurs a times,
2. the subsequence `01` occurs b times,
3. the subsequence `10` occurs c times, and
4. the subsequence `11` occurs d times.

However, Jingfei does not know how to create such a string – or whether it is even possible. Could you help her?

Input

The input consists of a single line with four integers a, b, c, and d ($0 \leq a, b, c, d \leq 10^9$).

Output

Output a bit string that satisfies the given requirements. If there are several solutions, output any one of them. If there are no solutions, output "`impossible`".

Sample Input 1	Sample Output 1
`3 4 2 1`	`01001`

Sample Input 2	Sample Output 2
`5 0 0 5`	`impossible`

This page is intentionally left (almost) blank.

Problem K
King's Colors
Problem ID: kingscolors
Time limit: 1 second

Picture by Geddo via Pixabay, CC0

Far, far away, there is the mystical Kingdom of Trees (more formally, "Royal Commonwealth of Connected Undirected Simple Acyclic Graphs"). The King there is very sad because his kingdom is not accepted as a sovereign state in the United Nations. In order to become a member, he needs to make a flag the UN can put on their website.

The flag will of course consist of the King's favorite tree, which contains n nodes. The King would be happy to just have the tree colored in black and white, but for the sake of his wife the Queen, he decided that the tree will contain all the favorite colors of their k children (they all have distinct favorite colors). Clearly, no two neighboring nodes can have the same color, but otherwise any coloring of the tree using exactly the k colors would make a feasible flag candidate.

How many different flags are possible?

Input

The first line contains two integers n and k ($2 \leq k \leq n \leq 2\,500$), where n is the number of nodes in the King's favorite tree and k is the number of children. Then follow $n-1$ lines describing the edges in the tree; the i'th of these lines contains a non-negative integer $p_i < i$, meaning that node p_i is the parent of i.

The nodes are numbered from 0 to $n-1$ and the tree is rooted at node 0. Note that the embedding of the tree on the flag is already fixed, the only thing that remains is to assign colors.

Output

Output the number of different possible color assignments. The number can be quite big, so the King has requested to know the answer modulo $1\,000\,000\,007$.

Sample Input 1	Sample Output 1
4 3 0 1 1	18

Sample Input 2	Sample Output 2
6 4 0 1 1 3 4	600