International Collegiate Programming Contest
2023

Latin American Regional Contests

October 21, 2023

Contest Session

This problem set contains 13 problems; pages are numbered from 1 to 21.

This problem set is used in simultaneous contests with the following participating countries:

Antigua y Barbuda, Argentina, Bolivia, Brasil, Chile, Colombia, Costa Rica, Cuba, El Salvador, Guatemala, México, Perú, Puerto Rico, República Dominicana, Trinidad y Tobago, Uruguay and Venezuela

v1.0

General information

Unless otherwise stated, the following conditions hold for all problems.

Program name

1. Your solution must be called *codename*`.c`, *codename*`.cpp`, *codename*`.java`, *codename*`.kt`, *codename*`.py3`, where *codename* is the capital letter which identifies the problem.

Input

1. The input must be read from standard input.
2. The input consists of a single test case, which is described using a number of lines that depends on the problem. No extra data appear in the input.
3. When a line of data contains several values, they are separated by *single* spaces. No other spaces appear in the input. There are no empty lines.
4. The English alphabet is used. There are no letters with tildes, accents, diaereses or other diacritical marks (ñ, Ã, é, Ì, ô, Ü, ç, etcetera).
5. Every line, including the last one, has the usual end-of-line mark.

Output

1. The output must be written to standard output.
2. The result of the test case must appear in the output using a number of lines that depends on the problem. No extra data should appear in the output.
3. When a line of results contains several values, they must be separated by *single* spaces. No other spaces should appear in the output. There should be no empty lines.
4. The English alphabet must be used. There should be no letters with tildes, accents, diaereses or other diacritical marks (ñ, Ã, é, Ì, ô, Ü, ç, etcetera).
5. Every line, including the last one, must have the usual end-of-line mark.
6. To output real numbers, round them to the closest rational with the required number of digits after the decimal point. Test case is such that there are no ties when rounding as specified.

Problem A – Analyzing Contracts

Doctor Kruskal is starting a tiberium trading business. They have N possible suppliers of tiberium, and many clients interested in receiving tiberium to run their own industries.

Calendar days are numbered chronologically using positive integers, and each supplier is identified by a distinct integer from 1 to N. Supplier i can supply tiberium on any day from day S_i onwards, but not on the days strictly before S_i. They charge a price of P_i dollars per day for such a service. Since Kruskal is very smart, the list of suppliers contains only the best suppliers in the city. Besides, it is the case that $S_i < S_{i+1}$ and $P_i > P_{i+1}$ for $i = 1, 2, \ldots, N-1$.

Kruskal's system keeps a database of available clients. Initially, this database is empty and contains no clients. Clients will be arriving one by one, and each of them is immediately added to the database upon arrival. The j-th client is interested in receiving tiberium on any day up to day E_j inclusive. For each day that they receive tiberium, their industry will generate R_j dollars of gross revenue. Thus, if Kruskal matches supplier i to client j, the final profit of this whole operation after deducting the tiberium cost will be $(R_j - P_i) \times (E_j - S_i + 1)$, where $S_i \leq E_j$, as otherwise no tiberium could be provided.

At any moment, Kruskal's system can quickly compute, for any particular supplier i, the optimal client among those in the database, so that the profit of the operation when matching the supplier and the client is maximized, and it can report such maximum profit. It might be the case that a positive profit for a supplier cannot be achieved with any of the available clients; in that case, the system reports a profit of zero.

Notice that when Kruskal's system is requested to compute the maximum profit for a given supplier, that supplier is matched with at most one of the available clients, and in that case, such a match has no effect at all on future operations. This means that both the supplier and the client can be considered again for future matchings.

Your task is to implement Kruskal's system.

Input

The first line contains an integer N ($1 \leq N \leq 2 \times 10^5$) indicating the number of suppliers.

The i-th of the next N lines describes supplier i with two integers S_i and P_i ($1 \leq S_i, P_i \leq 10^9$), denoting respectively the start day and the price per day for the supplier. It is guaranteed that $S_i < S_{i+1}$ and $P_i > P_{i+1}$ for $i = 1, 2, \ldots, N-1$.

The next line contains an integer Q ($1 \leq Q \leq 2 \times 10^5$) representing the number of operations that must be processed. Operations are described in the next Q lines, in the order they are executed in the system, one operation per line. There are two types of operations.

If the operation adds a client to the database, the line contains the lowercase letter "`c`", followed by two integers E and R ($1 \leq E, R \leq 10^9$), indicating respectively the end day and the gross revenue per day for the client.

If the operation requests to compute the maximum profit for a supplier, the line contains the lowercase letter "`s`", followed by an integer I ($1 \leq I \leq N$) that identifies the supplier. It is guaranteed that the input contains at least one operation of this type.

Output

Output a line for each operation of type "`s`". The line must contain an integer indicating the maximum possible profit when matching an available client with the given supplier. Write the results of the operations in the order they appear in the input.

Sample input 1	**Sample output 1**
4 2 8 4 5 7 3 9 2 11 s 1 c 10 10 s 1 s 2 s 3 s 4 c 7 26 s 2 s 4 s 3 s 1	0 18 35 28 16 84 16 28 108

Problem B – Blackboard Game

Carlinhos and Equalizer are playing a game. The game begins with $3N$ elements, which are integer numbers, written on a blackboard. Then, for N rounds, the following two steps are repeated.

1. Carlinhos, the first player, selects an unchosen element and marks it with a red circle.
2. Equalizer, the second player, picks two unchosen elements, marks one of them with a blue square, and erases the other from the blackboard.

At the end of these rounds, the blackboard contains N red-marked elements and N blue-marked elements, with no moves left. The game concludes with a clear winner: if the sum of the red-marked elements differs from the sum of the blue-marked elements, Carlinhos emerges victorious; otherwise, Equalizer takes the win.

The figure below depicts the only possible outcome for the first sample. In this case Equalizer wins for sure, no matter how they play both sums will be equal to 25.

5
5
5
5
5
5
5
5
5
5

Carlinhos, feeling the game is imbalanced, seeks to determine whether he can secure a victory when both players play optimally. Can you help him with this task?

Input

The first line contains an integer N ($1 \leq N \leq 1000$).

The second line contains $3N$ integers $B_1, B_2, \ldots, B_{3N}$ ($-10^5 \leq B_i \leq 10^5$ for $i = 1, 2, \ldots, 3N$), representing the numbers initially written on the blackboard.

Output

Output a single line with the uppercase letter "`Y`" if Carlinhos can win the game and the uppercase letter "`N`" otherwise, assuming both players play optimally.

Sample input 1	**Sample output 1**

`5`
`5 5 5 5 5 5 5 5 5 5 5 5 5 5 5`
	`N`

Sample input 2	**Sample output 2**

`2`
`1 2 4 8 16 32`
	`Y`

Explanation of sample 2:
Carlinhos wins no matter how he plays, since all subsets have distinct sums.

Sample input 3	**Sample output 3**

`1`
`2 3 3`
	`Y`

Explanation of sample 3:
Carlinhos can win by picking the number 2. Notice that he would have lost if he picked a 3.

Problem C – Candy Rush

It's rush hour! You've called it a day at work, and you need to buy candies for all your family members before the mall closes.

Exclusivity and uniformity are characteristics highly valued by your family, and in order to impress them, you've come up with a plan. All the candies given to each family member should be from a single brand, and no other family member should receive candies from that same brand. Additionally, you don't want to admit you love some more than others, so you want everyone to receive the same number of candies.

The mall has a store that sells candies from K different brands. Coincidentally, your family consists of exactly K members. This may seem too easy, but of course there's a catch.

The store displays its candies aligned on a single shelf. You don't have time to select candies individually; instead, you want to buy a group of *contiguous* candies to complete your task efficiently. This means that when you purchase any pair of candies, you must also buy all the candies located between them on the shelf.

What is the maximum number of candies that you can buy?

Input

The first line contains two integers N and K ($1 \leq N, K \leq 4 \times 10^5$), indicating respectively the number of candies on the shelf and the number of family members. Candy brands are identified by distinct integers from 1 to K.

The second line contains N integers $C_1, C_2, \ldots, C_N$ ($1 \leq C_i \leq K$ for $i = 1, 2, \ldots, N$), denoting the brand of each candy on the shelf, in left-to-right order.

Output

Output a single line with an integer indicating the maximum number of candies that you can buy to your family. Recall that no family member can receive two different candy brands, and no candy brand can be purchased for two different family members. Additionally, each family member must receive the same number of candies, and the candies must be purchased as a contiguous block from the shelf.

Sample input 1	**Sample output 1**

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

Explanation of sample 1:
Either buying from the first to the fourth candies or from the third to the sixth allows you to purchase two candies of each brand.

Sample input 2	**Sample output 2**

`7 3`
`2 1 2 1 2 2 3`
	`0`

Explanation of sample 2:
No contiguous block of candies has the same number of candies for brands 1, 2, and 3.

Sample input 3	**Sample output 3**

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

Explanation of sample 3:
While the store is known for selling candies from K different brands, some brands might be out of stock.

Problem D – Deciphering WordWhiz

WordWhiz is a popular word puzzle game that challenges players to guess a secret word within a limited number of attempts. The game uses a dictionary containing N words. Each word in this dictionary consists of five distinct lowercase letters.

The game begins with the player being presented with an empty grid, consisting of a number of rows. Each row allows a single guess. The player's task is to fill rows with words contained in the dictionary until the secret word is found, or the player has used all available rows.

After the player submits a guess, the game provides feedback by coloring the cells where the guess was written. The feedback consists of three colors:

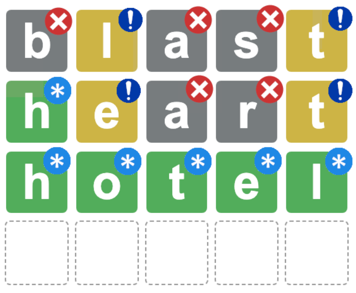
- Gray ("`X`"): The letter in the cell is not part of the secret word.
- Yellow ("`!`"): The letter in the cell is part of the secret word but is in the wrong position.
- Green ("`*`"): The letter in the cell is part of the secret word and is in the correct position.

To illustrate, let's consider the scenario where the secret word is "`hotel`", and the player submits "`blast`" as their guess. In this case, the first, third, and fourth cells would turn gray because "`b`", "`a`", and "`s`" are not present in the secret word "`hotel`". The second and fifth cells, however, would turn yellow. This indicates that "`l`" and "`t`" are part of the secret word but appear in wrong positions: "`l`" should be in the fifth position instead of the second, while "`t`" should be in the third position instead of the fifth. This feedback would be represented by "`X!XX!`".

Now, if the player submits "`heart`" as their guess, the third and fourth cells would still turn gray, because "`a`" and "`r`" are not in "`hotel`". The second and fifth cells would again turn yellow, because once more "`t`" is in the fifth position (instead of the third), and this time "`e`" is in the second position when it should be in the fourth. However, for this guess the first cell would turn green, indicating that "`h`" is the first letter in both the guess "`heart`" and the secret word "`hotel`". This feedback would be represented by "`*!XX!`".

Finally, if the player submits "`hotel`" as their guess, all cells would turn green since this is the secret word. This feedback would be represented by "`*****`".

The feedbacks above can be seen in the following picture.



Some time ago, your company added a WordWhiz player on its website and now wants to enhance the game by adding functionality to display previous game sessions. However, only the feedback for each guess was stored, not the submitted words. This means that it might not be possible to accurately recover the guesses submitted in each session, and before investing any further effort, the company wants to analyze the recorded game sessions.

Given a dictionary of five-letter words, the secret word (included in the dictionary) and the feedback for a game session, your task is to determine how many words in the dictionary could have been submitted as each guess.

Input

The first line contains an integer N ($1 \leq N \leq 1000$) indicating the number of words in the dictionary.

Each of the next N lines contains a string representing a word in the dictionary. All strings are different and each of them consists of five different lowercase letters. The first string is the secret word for the game session.

The next line contains an integer G ($1 \leq G \leq 10$) indicating the number of guesses during the game session.

Each of the next G lines contains a five-character string representing the feedback for a guess. The feedback string contains only the characters "`X`", "`!`" and "`*`", indicating respectively gray, yellow and green colors.

It is guaranteed that the input describes a valid game session.

Output

Output G lines, such that the i-th contains an integer indicating how many words in the dictionary could have been submitted on the i-th guess.

Sample input 1	**Sample output 1**
6 hotel weary heart blast pilot vague 3 X!XX! *!XX! *****	1 1 1

Explanation of sample 1:
The only possibility is that the player submitted guesses as described in the statement.

Sample input 2	**Sample output 2**
3 scale table maple 5 X!X** X!X** X!X** X!X** X!X**	2 2 2 2 2

Explanation of sample 2:
The feedback when either "`table`" or "`maple`" is submitted as a guess is "`X!X**`" (because the secret word is "`scale`"). This means that for this game session, the player could have submitted either of these words for each attempt.

Sample input 3	**Sample output 3**

```
4
scale
table
maple
smile
4
X!X**
*XX**
X!X**
*****
```
	```
	2
	1
	2
	1
	```

Sample input 4	**Sample output 4**

```
5
latin
mrica
think
solve
debug
1
*****
```
	`1`

Problem E – Elevated Profits

Marina, a digital influencer who loves traveling the world, is embarking on a promotional tour for a women's clothing brand called W^2M (From Woman to Woman Marina). Marina's journey takes her through N cities in Latin America, each with its unique charm, and identified with a distinct integer from 1 to N, called its popularity index.

To facilitate Marina's travels, W^2M has provided her with $N - 1$ transfers, connecting pairs of cities in a way that guarantees accessibility to all N cities. Marina can traverse these connections as many times as she pleases.

Marina's mission is to showcase the brand's dresses in each of the N cities, with a twist. Each time she visits a city for the first time, she must select a dress she hasn't worn before and capture the city's essence in a social media post. Every new picture she shares attracts followers, creating anticipation for the next one. The anticipation value for her first picture is 1, and it increments by 1 for each subsequent picture.

Marina can revisit any city as often as desired, but a new picture must only be posted on her initial visit to a city. Her goal is to maximize the profit of her tour, which is computed as the sum of the anticipation value of each picture multiplied by the popularity index of the city where the picture is taken. More precisely, let p_i be the popularity index of the city where the i-th picture is taken. With this information, the profit can be calculated as

$$\sum_{i=1}^{N} i \times p_i = 1 \times p_1 + 2 \times p_2 + \cdots + N \times p_N \ .$$

Now, Marina seeks your assistance. Given that the tour has to start in city $p_1 = R$, your task is to help Marina determine the maximum profit she can achieve by strategically planning the order of her city visits.

Input

The first line contains two integers N ($1 \le N \le 3 \times 10^5$) and R ($1 \le R \le N$), indicating respectively the number of cities and the initial city of the tour.

Each of the next $N - 1$ lines contains two integers U and V ($1 \le U, V \le N$ and $U \ne V$), indicating that there is a transfer between cities U and V. It is guaranteed that it is possible to reach every city by using the transfers.

Output

Output a single line with an integer indicating the maximum profit Marina can achieve on her promotional tour.

Sample input 1	**Sample output 1**

7 3
3 5
3 7
5 1
5 4
7 2
7 6
	121

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

Problem F – Forward and Backward

A distant planetary system has a single sun and $N - 1$ planets. Each planet is identified by a distinct integer from 2 to N. In planet b, numbers are represented using base b.

A palindromic number is a number that remains the same when its digits are written both forward and backward. In this context, leading zeroes are not considered when determining if a number is palindromic.

The same number can be palindromic in one planet's base but not in another. For instance, in base 10, the number 33 is palindromic. It is also palindromic in base 2 and base 32 but not in bases such as 3 or 33, since $33_{10} = 100001_2 = 1020_3 = 11_{32} = 10_{33}$.

The inhabitants of this planetary system have a peculiar fondness for palindromic numbers and want to know which planets make the number N a palindromic number when represented in their base. Your task is to help them with this cosmic challenge.

Input

The input consists of a single line that contains an integer N ($2 \leq N \leq 10^{12}$) indicating the number to be checked for palindromic representation. N is given in base 10.

Output

Output a single line with an increasing list of integers in the interval $[2, N]$, indicating the planets in which N is a palindromic number when expressed in the base of the planet's identifier. Output these integers in base 10. If N is not palindromic in any of the planets, output the character "*" (asterisk) instead.

Sample input 1	**Sample output 1**
`33`	`2 10 32`

Sample input 2	**Sample output 2**
`3`	`2`

Sample input 3	**Sample output 3**
`2`	`*`

Problem G – GPS on a Flat Earth

On the day when aliens finally attacked humanity, nobody could have anticipated their weapon of choice. No nuclear weapons, meteors, lasers, or giant monsters. Instead, our planet was subjugated with the power of physics!

Specifically, the aliens transformed Earth into a two-dimensional, flat surface, forever neutering our space-faring capabilities. Although frustrated, humanity survived, and we resumed our lives as best as we could. This new two-dimensional existence requires many adjustments, including the use of GPS (Global Positioning System).

GPS normally works by using radio waves to measure the Euclidean distances from the user to several reference points (satellites), and using these distances to calculate the user's coordinates. However, the now flat Earth has two quirks we need to adapt to:

- Without satellites in orbit, we need to use radio towers instead. Each radio tower now has coverage over the entire planet due to the flat surface.
- Radio waves, which propagate differently in a two-dimensional world, require a shift from Euclidean to Manhattan distance for accurate calculations. Given any two points (X_1, Y_1) and (X_2, Y_2), the Manhattan distance between them is defined as $|X_1 - X_2| + |Y_1 - Y_2|$.

Your task is to write software for these adapted GPS calculations. Given a list of locations of N reference radio towers and their respective Manhattan distances to the GPS user, your algorithm must provide a list of possible locations of the user. These potential user locations are limited to those that are exactly at the measured Manhattan distance from each reference radio tower. The GPS is still in the initial test phase, so the user's true location is limited to integer coordinates.

Input

The first line contains an integer N ($1 \le N \le 10^5$) indicating the number of reference radio towers.

Each of the next N lines describes a tower with three integers X, Y ($-10^4 \le X, Y \le 10^4$), and D ($0 \le D \le 4 \times 10^4$), representing that a tower with coordinates (X, Y) is at Manhattan distance D from the GPS user. No two towers have the same location. It is guaranteed that the input data is reliable, pinpointing a non-empty finite set of possible locations for a user with integer coordinates.

Output

Output several lines. Each line must contain a different pair of integers X_u and Y_u indicating that (X_u, Y_u) is a user location compatible with the input data. The lines must be sorted by non-decreasing X_u value, breaking ties by increasing Y_u value.

Sample input 1	**Sample output 1**

```
2
1 1 5
7 0 4
```
	```
	4 -1
	5 2
	```

Sample input 2	**Sample output 2**

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

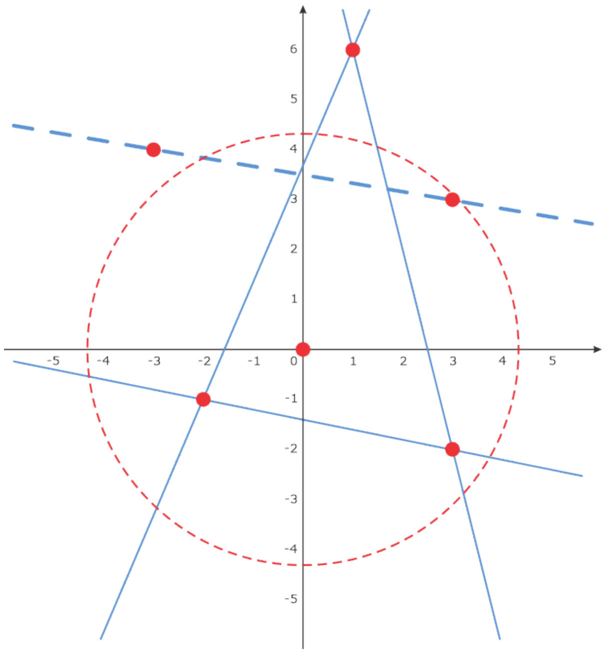
Problem H – Health in Hazard

In the frozen wilderness, a polar bear makes its home on a vast glacier, represented as a 2D plane. The coordinates of the bear's den are $(0, 0)$. To be healthy, each time the bear wakes up in its den, it walks to any point at distance exactly D from the den (measured using Euclidean distance).

With the bear facing the challenges of a changing climate, a team of dedicated scientists and mathematicians has set out to assist. They have received a detailed report filled with predictions of the impending impact of global warming on the glacier in the coming days. The report contains the predictions in chronological order, and each of them is represented by an infinite line that corresponds to a melting event. After each prediction comes true, the line that represents it can no longer be crossed by the bear.

Initially, the glacier is considered to be infinitely large in all directions, and the bear can roam freely. However, as the team of scientists and mathematicians, you understand the bear's dilemma: according to the predictions, the glacier might eventually shrink in such a way that the bear can no longer be healthy. Your task is to calculate the earliest moment in which this will happen, that is, when no point at distance exactly D from the den can be reached by the bear.

The figure below depicts the first sample. The circumference contains the points at distance exactly D from the den. When just the first three predictions are considered (solid lines), the bear can still reach points in the circumference. Once the fourth prediction (dashed line) is also considered, no point in the circumference can be reached from the den.



Input

The first line contains an integer N ($1 \leq N \leq 2 \times 10^5$) and a rational number D with at most five digits after the decimal point ($1 \leq D \leq 10^6$). The value N indicates the number of predictions, while D represents the distance from the den.

Each of the next N lines describes a prediction with four integers X_1, Y_1, X_2 and Y_2 ($-10^6 \leq X_1, Y_1, X_2, Y_2 \leq 10^6$ and $(X_1, Y_1) \neq (X_2, Y_2)$), which define an infinite line in the plane passing through (X_1, Y_1) and (X_2, Y_2). Each prediction indicates that the corresponding line can no longer be crossed by the bear. Predictions are given in chronological order and they are identified by distinct integers from 1 to N, according to that order. It is guaranteed that no prediction defines a line passing through the den.

Output

Output a single line with an integer identifying the earliest prediction that indicates that no point at distance exactly D from the den can be reached by the bear. If this situation never occurs, output the character "`*`" (asterisk) instead. It is guaranteed that variations in the value of D within a range of $\pm 10^{-5}$ from the value given in the input do not alter the output.

Sample input 1	**Sample output 1**

`5 4.321`
`-2 -1 3 -2`
`1 6 3 -2`
`1 6 -2 -1`
`-3 4 3 3`
`-2 1 5 4`
	`4`

Sample input 2	**Sample output 2**

`5 2`
`1 0 1 1`
`-1 0 -1 -1`
`3 1 1 3`
`1 3 3 1`
`0 4 4 0`
	`*`

Problem I – Inversions

Every year, mathematicians and computer scientists from around the globe gather for the prestigious Inversion Counting Puzzle Contest (ICPC). For the next ICPC, the organizers had prepared the following challenge: given a string S consisting of lowercase letters, count the number of *inversions* in it. An inversion is a pair of indices $i < j$ such that S_i (the letter at position i) comes after S_j in the alphabet.

However, just last month, a group of outstanding researchers devised a sophisticated algorithm that can count the inversions in a string extremely fast. While this was great news for the advancement of science, it has been a nightmare for the ICPC staff, since their planned challenge is now obsolete.

This issue escalated to the head problem setter, who then presented a clever idea. Instead of simply receiving a string S, they should ask participants to repeat this string N times before counting the inversions. If the judges then set N to be large enough, at some point the algorithm proposed by the researchers will start to be too slow. Happy with this idea, the ICPC staff went ahead with organizing the next contest.

Unfortunately, now the judges don't know the answers to the input files anymore, and therefore can't judge submissions! The ICPC has just kicked off, and participants are about to start submitting their solutions. Please help the judges by computing the answers, so that the ICPC can run smoothly.

Input

The first line contains a string S ($1 \le |S| \le 10^5$), which is made of lowercase letters.

The second line contains an integer N ($1 \le N \le 10^{12}$) indicating how many times the string S is to be repeated.

Output

Output a single line with an integer indicating the number of inversions in the string S^N (S repeated N times). Because this number can be very large, output the remainder of dividing it by $10^9 + 7$.

Sample input 1	**Sample output 1**

`ba`
`1`
	`1`

Sample input 2	**Sample output 2**

`ab`
`3`
	`3`

Sample input 3	**Sample output 3**

`zkba`
`1`
	`6`

Sample input 4	**Sample output 4**

`cab`
`7`
	`77`

Problem J – Journey of the Robber

Monopolis is a beautiful and wealthy country. Among its impressive features is the layout of the country, where N cities are interconnected by $N-1$ roads of equal length, allowing to travel between any two cities.

Another unique feature of Monopolis is that each city has a single bank with a different amount of money. Thus, it is possible to assign a distinct number from 1 to N to each city, representing its wealth ranking relative to the other cities, with city 1 having the least money and city N having the most.

Rob is planning a "business trip" to Monopolis. Rob's business is, in fact, robbing. Robbing banks, to be more precise. Rob is an ambitious robber and follows a particular modus operandi: he only targets banks with more money than the one he just robbed. Thus, after robbing in a city i, he moves to the closest city having more money. If there are multiple cities with more money than i at the same distance, he selects the one with the least money. If there's no city richer than i, he remains in the same city and reflects on his actions.

Even though Rob is very set on his modus operandi, he is still planning his business trip to Monopolis, and then he asks for your assistance. Rob wants to know for each city i which would be the next city to visit in case his first heist is at city i.

Input

The first line contains an integer N ($1 \le N \le 10^5$) representing the number of cities in Monopolis. Each city is identified by a distinct integer from 1 to N, in increasing order of wealth.

Each of the next $N-1$ lines contains two integers U and V ($1 \le U, V \le N$ and $U \ne V$), indicating that there is a two-way road between cities U and V. It is guaranteed that there is a path between each pair of cities using the given roads.

Output

Output a single line with N integers, such that the i-th of them indicates the next city Rob would visit in case his first heist is at city i.

Sample input 1

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

Sample output 1

```
6 5 5 5 6 6
```

Sample input 2

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

Sample output 2

```
3 3 4 5 5
```

Sample input 3

```
1
```

Sample output 3

```
1
```

Problem K – Keen on Order

Nloglonia is hosting a movie festival that spans N days. There are K distinct movies, and on each of the N days, one of them will be displayed. Each of the available movies might be displayed on multiple days or not displayed at all during the festival.

The festival schedule is given as an integer array V of size N, with $1 \leq V_i \leq K$, indicating which movie will be shown on each day. Bob wants to watch all K movies, and he firmly believes that the order in which he watches them will significantly affect his experience. So now he wonders: is it true that for every order of the movies, he would be able to pick K days to visit the festival and watch those movies in that given order? More formally, is it true that every permutation of $1, 2, \ldots, K$ is a subsequence of V? If this is not the case, Bob also wants you to find some permutation (any one) that is not.

Input

The first line contains two integers N and K ($1 \leq N, K \leq 300$), indicating respectively the number of days the movie festival lasts and the number of movies available for display.

The second line contains N integers $V_1, V_2, \ldots, V_N$ ($1 \leq V_i \leq K$ for $i = 1, 2, \ldots, N$), indicating that the movie V_i will be displayed on day i.

Output

Output a single line with K integers showing a permutation of $1, 2, \ldots, K$ which is not a subsequence of V. If every permutation is a subsequence of V, output the character "*" (asterisk) instead.

Sample input 1	**Sample output 1**

`9 3`
`1 2 3 1 2 3 1 2 3`
	`*`

Sample input 2	**Sample output 2**

`11 4`
`1 2 3 4 2 3 3 2 4 1 4`
	`3 4 1 2`

Sample input 3	**Sample output 3**

`11 4`
`1 2 3 4 2 3 3 2 4 1 4`
	`4 1 2 3`

Sample input 4	**Sample output 4**

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

Problem L – Latam++

The world does not have enough programming languages yet. To help with that, the Internal Committee for the Perfection of C (ICPC) is planning to build a brand new programming language: `Latam++`.

In `Latam++`, a variable name consists exclusively of one or more lowercase letters of the English alphabet. A valid expression is a "well-formed" string, expressing how to combine variables by using the four arithmetic binary operators "`+`", "`-`", "`*`" and "`/`", possibly with parentheses.

Formally, valid expressions are exactly those strings that can be produced by the following rules.

- A variable name is a valid expression.
- Surrounding any valid expression in parentheses produces another valid expression.
- If A and B are valid expressions, then the concatenation AcB is a valid expression, where c is any of the four arithmetic binary operators "`+`", "`-`", "`*`" and "`/`".

Thus, the following are all valid expressions:

- `a+b`
- `a+b*(c+b)`
- `atoms+boots*(charly+bob)`
- `(((a)))*(bbasdsaqwe/a/a/a)`

On the contrary, the following are not valid expressions:

- `a+`
- `a+b(c+b)`
- `atoms+boots*((charly+bob)`
- `((()))*(bbasdsaqwe/a/a/a)`

The language is far from complete, and it will likely take ICPC decades of debates until the first version of `Latam++` is released. Meanwhile, we will focus only on a specific and very special feature of its compiler, called Automatic Valid Substring Expression Counting (AVSEC).

AVSEC is an extremely useful feature, where the compiler reports the total number of substrings of a given string that are valid expressions. Your task is to implement AVSEC.

For counting purposes, two substrings are considered different if they start or end at different indexes, even if the corresponding strings are identical (that is, they are the same sequence of characters).

Input

The input consists of a single line that contains a string S ($1 \leq |S| \leq 2 \times 10^5$), which is made up of lowercase letters, opening or closing parenthesis, and the four characters "`+`", "`-`", "`*`" and "`/`".

Output

Output a single line with an integer indicating the number of substrings of S that are valid expressions.

Sample input 1	**Sample output 1**
`a+b(c+b)`	`7`

Explanation of sample 1:
The seven substrings are “`a`”, “`a+b`”, “`b`” (third character), “`(c+b)`”, “`c`”, “`c+b`”, and “`b`” (seventh character).

Sample input 2	**Sample output 2**
`aa`	`3`

Sample input 3	**Sample output 3**
`a-a`	`3`

Problem M – Meeting Point

Your friend Pedro always gets very excited for group activities. In his excitement, he runs so fast to the meeting point that he gets tired before arriving. One day, you decided to gather data on this phenomenon and, surprisingly, noticed that he consistently gets tired exactly at the midpoint of his route. In other words, he gets tired when he has traveled half the distance he was going to travel.

Your city has N crossroads identified by distinct integers from 1 to N, and M two-way roads. Each road has a length and connects a specific pair of crossroads, in such a way that there is a path in the city between every pair of crossroads. The distance between two crossroads is the length of a minimum path between those crossroads.

Pedro lives at crossroad P, and your group of friends decided to meet at crossroad G later today. After thinking for a while, you devised the following plan so that Pedro arrives on time. You will tell him a misleading meeting point so that he gets tired exactly at G. To make this plan work, crossroad G must belong to every path that Pedro could possibly take while going from P to the misleading meeting point, and for each such path, Pedro must get tired exactly at G. Fortunately, you know that Pedro is a good planner and would never take a path longer than needed.

Now you wonder, which crossroads would work as that misleading meeting point?

Input

The first line contains two integers N ($2 \leq N \leq 10^5$) and M ($1 \leq M \leq 10^5$), indicating respectively the number of crossroads and the number of roads in the city. Each crossroad is identified by a distinct integer from 1 to N.

The second line contains two integers P and G ($1 \leq P, G \leq N$ and $P \neq G$), denoting respectively the crossroad where Pedro lives and the correct meeting point.

Each of the next M lines describes a road with three integers U, V and D ($1 \leq U, V \leq N$, $U \neq V$ and $1 \leq D \leq 10^9$), representing that there is a two-way road of length D between crossroads U and V.

It is guaranteed that there is a path between each pair of crossroads using the given roads, and there is at most one road between each pair of crossroads.

Output

Output a single line with an increasing list of integers indicating the crossroads that would work for your plan. If no crossroad would work, output the character "`*`" (asterisk) instead.

Sample input 1	**Sample output 1**

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

Sample input 2	**Sample output 2**

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

Sample input 3	**Sample output 3**

`3 2`
`1 2`
`1 2 100000`
`2 3 99999`
	`*`

Sample input 4	**Sample output 4**

`4 4`
`4 3`
`3 4 1`
`4 1 1`
`1 2 1`
`2 3 1`
	`*`