Problem A. A Colorful Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Luiz, Carol, and Henrique agreed to meet at the park after the UnBalloon running club meeting to organize their team for the VII UnBalloon Marathon, the most anticipated race of the year. However, Luiz, being an excellent runner, finished the course before the rest of the team, coming in 3^{rd} place in the club (behind Maxwell and Arthur).

While waiting for Carol, who is slower due to a knee injury, and Henrique, who is still waiting for the bus, Luiz began observing the trees in the park. He noticed that one tree was quite different from the others, having several parts of distinct colors. Knowing that if he mentioned this, his team would rather investigate the colors instead of focusing on the upcoming marathon, help Luiz answer his teammates' various questions.

The tree observed by Luiz can be modeled as a connected acyclic undirected graph with N nodes and $N-1$ edges, where the root is fixed at vertex 1. Each node i has an associated color, represented by an integer value a_i.

Your task is to help the team answer Q queries. Each query consists of a node x_i, and you must determine the number of distinct colors present on the unique path between node x_i and the root of the tree.

Input

The first line contains two integers N and Q ($1 \le N, Q \le 10^5$), representing the number of nodes in the tree and the number of queries, respectively.

The second line contains N integers $a_1, a_2, \ldots, a_N$ ($1 \le a_i \le 10^9$), where a_i denotes the color of the i-th node.

The following $N-1$ lines each contain two integers u_i and v_i ($1 \le u_i, v_i \le N, u_i \ne v_i$), representing an edge between nodes u_i and v_i.

Finally, the next Q lines each contain a single integer x_i ($1 \le x_i \le N$), representing a query for the number of distinct colors on the path from node x_i to the root.

Output

For each query, print a single line containing a single integer representing the number of distinct colors found on the unique path between node x_i and the root of the tree.

Examples

standard input	standard output
4 2 1 2 3 4 1 2 2 3 3 4 3 4	3 4
4 2 1 2 3 2 1 2 2 3 3 4 3 4	3 3

Problem B. Beats

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Nirva is one of Unballoon's newest talents. As everyone knows, one of the main pieces of advice we hear at the beginning of a competitive programming career is to focus on improving reasoning skills and leave learning more advanced topics, such as suffix automaton, flow, FFT, etc., for later. As everyone also knows, this advice is almost always completely ignored. Because of that, Nirva decided to learn "Segment Tree Beats". The author of this data structure himself has the following to say about it:

"In China, all of the 15 candidates for the Chinese National Team are asked to write a simple research report about algorithms in informatics Olympiad, and the score will be counted in the final selection. There are many interesting ideas and algorithms in these reports. And I find that some of them are quite new for competitors in CF although they are well known in China [...].

This blog is about my report which is written about two years ago. [...] the name "Segment tree beats" is given by C_SUNSHINE which is from a famous Japanese anime Angel Beats."

As expected, Nirva easily understood how Segment Tree Beats works and made her own implementation. To test whether it is correct, she thought it would be interesting to compare her implementation with yours.

You are given n non-negative integers $a_1, \ldots, a_n$. Your program must process q operations of three types:

1. chmod(l, r, x) — For all indices i such that $l \le i \le r$, replace the value of a_i with $a_i \,\%\, x$.
2. chmax(l, r, x) — For all indices i such that $l \le i \le r$, replace the value of a_i with $\max(a_i, x)$.
3. sum(l, r) — Print the sum of the elements a_i such that $l \le i \le r$, that is, $a_l + a_{l+1} + \cdots + a_r$.

Input

The first line contains two integers n and q ($1 \le n, q \le 1000$). The second line contains n integers $a_1, \ldots, a_n$ ($0 \le a_i \le 10^6$).

Each of the next q lines describes an operation in one of the following formats:

- $1\ l\ r\ x$ — representing the operation chmod(l, r, x);
- $2\ l\ r\ x$ — representing the operation chmax(l, r, x);
- $3\ l\ r$ — representing the operation sum(l, r).

It is guaranteed that, in each operation, $1 \le l \le r \le n$ and $1 \le x \le 10^6$.

Output

For each operation of type 3 (sum), print a single integer representing the result of that operation.

Example

<table>
<tr><th>standard input</th><th>standard output</th></tr>
<tr><td><pre>6 5
1 2 3 4 5 6
3 1 5
1 2 4 3
3 1 5
2 1 4 2
3 2 3</pre></td><td><pre>15
9
4</pre></td></tr>
</table>

Problem C. Chasing the Treasure

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Tiagodfs, a legendary competitive programmer, decided to hide his most valuable solutions in a rectangular field of dimensions $b \times h$.

The hiding spot was chosen completely at random: every point inside the field has the same probability of containing the treasure.

After a long investigation, the team *Chappell Roan, vc tá jurada aqui em São Sebastião* finally discovered the existence of this hiding spot. To recover the secret codes, they decided to use a special excavation machine capable of digging a region in the shape of a regular hexagon with side length l. The excavated region will be positioned so that it is entirely contained within the field.

Determine the probability that the team *Chappell Roan, vc tá jurada aqui em São Sebastião* finds the treasure with this kind of excavation.

Input

The input contains three integers b, h, and l ($1 \leq b, h, l \leq 10^3$).

It is guaranteed that some regular hexagon of side length l fits entirely inside the rectangle.

Output

Print the probability that the treasure lies inside the excavated region.

Your answer will be considered correct if the absolute or relative error does not exceed 10^{-4}.

Examples

standard input	standard output
`10 10 5`	`0.6495190528`
`29 23 11`	`0.4713151748`

Problem D. Dominating Shurikens

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

In the Hidden Village of Balloons, those who seek to improve their skills are always close to the training field. The field area is vast, with a section containing N logs lined up, all N meters tall, numbered from 1 to N. To use them as a kind of scale, each log numbered i was marked exactly every i meters (up to N). For example: if $N = 3$, log 1 has marks at 1, 2, and 3 meters, log 2 has a mark at 2 meters, and log 3 has a mark at 3 meters.

Among the ninjas in the training field on a certain day was Lucas Sala — a shuriken specialist, capable of throwing more than one billion shurikens simultaneously! He was seeking to perfect his Wind Style: Multiple Aerial Shurikens jutsu, which can hit several opponents with a large number of shurikens. To do this, he was using the logs in the training field as follows: he chose two integers l and r, with $1 \le l \le r \le N$, and threw x shurikens at each of the logs from l to r inclusive — hitting them all, of course.

However, Lucas Sala's feat was not enough to impress his sensei, José. So he proposed a challenge for his student: José chose integers l, r, and k ($1 \le l, r, k \le N$), and, among the logs from l to r, Lucas Sala should throw x shurikens only at those with exactly k marks. This challenge was performed several times, with different values of l, r, and k. However, José was not able to keep track of all the shurikens thrown to verify Lucas Sala's aim, so he stopped at some moments between challenges to choose other integers l and r and count the number of shurikens hit on each log from l to r.

Thus, being very careful, José wrote down in his scroll two types of events related to that day's training, in the order they occurred:

1. $l\ r$: José counts the total number of shurikens hit on all logs from l to r.
2. $l\ r\ k\ x$: Lucas Sala hits x shurikens on every log i such that $l \le i \le r$ and log i has exactly k marks.

That day was many years ago, and José remembers that time fondly. Looking at his scroll, he realizes that he did not write down, for the type 1 events, the total number of shurikens he counted, which is a pity. Help him relive the old days and recover that information for him!

Since each counted quantity can be very large, you must print the remainder of each one when divided by $10^9 + 7$.

Input

The first line contains two numbers: N ($1 \le N \le 10^{12}$), the number of logs and also the height of each one of them, in meters, and Q ($1 \le Q \le 10^5$), the number of events that occurred during the training.

Each of the next Q lines contains an event, in one of the following formats:

- $1\ l\ r$ ($1 \le l \le r \le N$), representing an event in which José counted shurikens.
- $2\ l\ r\ k\ x$ ($1 \le l \le r \le N, 1 \le k \le N, 0 \le x < 10^9 + 7$), representing an event in which Lucas Sala threw shurikens.

Output

The output must consist of one line for each event of the first type, each containing a single number x ($0 \le x < 10^9 + 7$): the remainder, upon division by $10^9 + 7$, of the number of shurikens counted by José in the respective event.

Examples

<table>
<tr><th>standard input</th><th>standard output</th></tr>
<tr><td>5 4
2 1 5 1 10
2 2 4 2 5
1 1 5
1 2 3</td><td>35
15</td></tr>
<tr><td>10 7
2 1 10 2 1000000000
2 1 10 2 1000000000
1 4 5
2 1 7 1 50
1 6 8
2 8 10 5 100
1 2 2</td><td>999999979
100
0</td></tr>
<tr><td>1000000000000 2
2 1 1000000000000 2 5
1 400000000000 600000000000</td><td>999996512</td></tr>
</table>

Problem E. EZ

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

The secret society UnBalloon has many members, many contacts, and many agreements. Because of a bet lost in rock-paper-scissors during a trip, UnBalloon became responsible for sending, every full moon, a supply of mice to feed the forest witch's pet snake.

In a certain month, the full moon fell on Wednesday, Caleb's day off from work (one of UnBalloon's members). Because of this, he was chosen to deliver the supply of mice, instead of drinking Martinis at the bar as usual. Being very methodical, Caleb decided to take N mice, where the i-th heaviest of these mice weighs exactly i kilograms.

When entering the witch's cottage to make the delivery, he heard her saying something about "...not giving wings to the snake...". Perhaps because of that, as soon as she received the supply of mice, she enchanted them as follows: "After eating a mouse weighing i kilograms, it will not be possible to eat mice weighing $i-1$ or $i+1$ kilograms.".

Caleb was astonished by this and thought: "What will then be the greatest total amount, in kilograms, that the snake will be able to eat under this restriction?". To his surprise, the snake heard his thoughts and replied (telepathically): "EZ...". Caleb knows that EZ is an abbreviation of easy. But is eating the greatest possible amount really that easy?

Calculate for Caleb the greatest total amount, in kilograms, that the snake will be able to eat, considering the restriction imposed by the witch's spell.

Input

The input consists of a single line containing a single integer N: the number of mice ($1 \leq N \leq 1000$).

Output

Print a single line containing a single integer: the greatest total amount, in kilograms, that the snake can eat in total.

Examples

standard input	standard output
1	1
3	4
4	6
1000	250500

Note

In the first test case, with only one mouse, the snake will only eat it (total: 1 kilogram).

In the second test case, the snake can eat mice weighing 1 and 3, resulting in 4 kilograms eaten (which is the maximum possible).

In the third test case, the snake can eat mice weighing 2 and 4, resulting in 6 kilograms eaten (which is the maximum possible).

Problem F. Forms of Forming Teams

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

The secret organization UnBalloon has N agents, numbered from 1 to N. Using a secret algorithm, the organization calculated, for each of them, their value as a human being, so they could order the agents by this value (no two agents have the same value). From this, the list L was created, where L_i is the number of the i-th most valuable agent.

The organization must form a team of agents for the next mission, which is to infiltrate the Super Ultra Secret Pizza. But for the mission to be successful, the chosen agents must follow a special property. Let S be the set of numbers of the chosen agents and P be the set of positions of these agents in the list L. From these sets, we define the polynomials[1] $A(x) = \sum_{i \in S} x^i$ and $B(x) = \sum_{i \in P} x^i$. For this team of agents to be successful, it is necessary that S is not an empty set and that there exists some polynomial $C(x)$ with all its coefficients being non-negative integers such that $A(x) \cdot C(x) = B(x)$.

As part of the organization's analysts, you must calculate how many ways there are to form a team of agents that would be successful in the mission. Since this quantity can be very large, calculate the remainder of this quantity when divided by 998244353.

[1] A polynomial is a function of the form $P(x) = \sum_{i=0}^{\infty} a_i \cdot x^i$ such that the number of non-zero values a_i is finite. The a_i's are called coefficients.

Input

The first line of input contains the integer N: the number of agents ($1 \le N \le 5 \cdot 10^3$).

The second line of input contains N distinct integers: the list L ($1 \le L_i \le N$).

Output

Print, on a single line, the remainder when dividing by 998244353 of the number of ways to form a team of agents for a successful mission.

Examples

standard input	standard output
1 1	1
4 2 1 4 3	6
4 1 3 2 4	8
10 1 4 5 3 2 10 9 8 7 6	49

Problem G. Great Contest

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

During the preparation for the UnBalloon Great Programming Contest, several coaches are testing strategies to assemble the best possible superteam.

The organizers of the event allow the formation of a superteam from a subset of n available teams. Each team has four types of participants, corresponding to different skill profiles, with their quantities within team i being represented by four integers (a_i, b_i, c_i, d_i). To assemble a super team, the organizers can choose any subset of these teams.

After choosing a superteam, the organizers gather all participants from the selected teams and define:

- A as the total number of participants of type a;
- B as the total number of participants of type b;
- C as the total number of participants of type c;
- D as the total number of participants of type d.

Then, a training session is held with the members of the superteam, with an associated numeric penalty. The training consists of two independent stages:

- In the first stage, participants of types a and b in the superteam are paired against each other for a round of confrontations. Each participant faces exactly one opponent whenever possible. At the end, exactly $|A - B|$ participants remain unmatched, which will contribute to the penalty calculation;
- In the second stage, the same process occurs between participants of types c and d.

Immediately after these two stages, two parameters (k, l) are used to calculate the penalty of the training. The values k and l represent the weight of each training stage, and the penalty value is given by the sum of the contributions of both stages.

- The contribution of the first stage is the number of participants remaining after the confrontations between types a and b, multiplied by k.
- The contribution of the second stage is the number of participants remaining after the confrontations between types c and d, multiplied by l.

The coaches want to test different strategies. For this purpose, they will evaluate q different pairs of weights (k, l).

Your task is to help the coaches: for each pair of weights (k, l), determine the **maximum penalty** that could be achieved by choosing a subset of the teams.

Input

The first line contains two integers n and q $(1 \leq n, q \leq 10^5)$ — the number of teams and the number of queries.

Each of the next n lines contains four integers a_i, b_i, c_i, d_i $(0 \leq a_i, b_i, c_i, d_i \leq 10^4)$, describing the i-th team.

Each of the next q lines contains two integers k and l $(0 \leq k, l \leq 10^4)$, representing a pair of weights.

Output

For each of the q pairs of weights, print one integer: the maximum possible penalty that can be achieved.

Examples

standard input	standard output
2 3 5 3 4 1 6 1 3 4 1 1 2 0 0 3	9 14 9
5 4 5 10 0 3 9 3 2 3 7 4 2 2 2 8 5 6 4 12 9 8 1 1 3 1 9 2 7 4	22 60 177 145

Problem H. Hours in Class

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 256 megabytes

AnaLu spent the whole day at LINF studying Number Theory, so she decided to put her new knowledge into practice! Because of that, when she got home, she went straight to her room to solve problems. Then, after many ACs, she ended up falling asleep. And she had a dream...

In this dream, AnaLu was once again a 6th-grade student, and the class of the day was about multiplication, exponentiation, and divisibility. The teacher wrote the following definitions on the board:

1. Consider positive integers A and X.
2. $A \cdot X$: A multiplied by X.
3. A^X: A raised to the power X (which is the same as doing $A \cdot A \cdot A \cdot \ldots \cdot A$, with X factors equal to A).
4. $A|X$: means that A divides X, that is, there exists some positive integer B such that $A \cdot B = X$.

But, not satisfied, the teacher decided to challenge her students. She asked each one to keep in their notebook the value of the number A, which initially is equal to 1. The teacher will make Q **queries** about this number A, in which a number X will be written on the board and all students must either update the value of A accordingly or answer a question, depending on the type of the query. There are three types:

1. All students must update the value of A to $A = A \cdot X$ in their notebook.
2. All students must update the value of A to $A = A^X$ in their notebook.
3. All students must answer (in English, with "Yes" or "No"): $X|A$?

AnaLu then realized that she was in a dream, since, depending on the teacher's queries, the number A could become so large that it would be impossible to write it down even if all the notebooks in the universe were put together. Even so, she realized that it would still be possible to answer the teacher's divisibility queries (type 3) correctly with the help of a computer.

But AnaLu did not bring any computer into the dream, so you must help her answer the queries!

Input

The first line contains an integer Q ($1 \leq Q \leq 10^5$), the number of queries.

The next Q lines contain two integers t ($1 \leq t \leq 3$) and X ($2 \leq X \leq 10^5$), the type of the query and its parameter. It is guaranteed that there is at least one query of type 3.

Output

For each query of type 3, print a line with the answer to that query: "Yes" if X is a divisor of A, or "No" otherwise.

Example

standard input	standard output
6 2 100000 1 3 2 3 1 2 3 12 3 18	No Yes

Note

Explanation of the example:

Initially, $A = 1$.

After the first query, $A = 1^{100000} = 1$.

After the second query, $A = 1 \cdot 3 = 3$.

After the third query, $A = 3^3 = 27$.

After the fourth query, $A = 27 \cdot 2 = 54$.

For the fifth query, 12 does not divide A, so the answer to the query is "No".

For the sixth query, 18 divides A, so the answer to the query is "Yes".

Problem I. Inventing Communications

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

Iasmim, Pedro, and Emerson are preparing to present their own encryption method to the *International Committee for Poor Communication* (ICPC), the leading organization that studies poor communication methods that "work".

They have developed a unique and innovative method for transmitting messages. Since the idea is to create a bad communication method, they send an encrypted sequence of numbers. To decrypt and find the original message, one must find the lexicographically smallest non-empty subsequence of the vector V such that its hash is strictly less than the hash of its complement. If a subsequence (of size k) consists of the positions $p_0, p_1, \ldots, p_{k-1}$, its complement is the subsequence composed of all the positions in the vector that are not any of these k positions.

We say that a sequence A is lexicographically smaller than a sequence B if, at the first position where they diverge, the element of A is smaller than the element of B. If the sequences are identical until the shorter one ends, the sequence of shorter length is considered the lexicographically smaller one.

The hash calculation used by the trio follows a polynomial hash structure. Given a subsequence (or its complement) composed of the numbers $x_0, x_1, \ldots, x_{k-1}$ in the order in which they appear in the vector V, the hash value H is calculated using the following formula:

$$H = (x_0 \cdot P^N + x_1 \cdot P^{N-1} + \ldots + x_{k-1} \cdot P^{N-k+1})\%M$$

Where:

- x_i is the value of the i-th element of the subsequence;
- $P = 100019$ is the polynomial base;
- N is the total length of the vector V;
- $M = 998244353$ is the modulus;
- The index i (from 0 to $k-1$) represents the position of the number within the chosen subsequence.
- By convention, the hash of an empty sequence ($k = 0$) is defined as 0.

You will be given the vector V and must recover the subsequence that represents the message communicated by Iasmim, Pedro, and Emerson.

Input

The first line of the input contains a single integer N ($1 \le N \le 10^6$), the size of the vector V.

The second line of the input contains N integers V_i: the elements of the vector V ($1 \le V_i \le 10^5$).

It is guaranteed that both N and the vector V are generated randomly for all hidden test cases.

Output

Print, on a single line, the lexicographically smallest non-empty subsequence A of the vector V such that $H(A) < H(A^c)$, where A^c is the complement of A in V. If no subsequence satisfies this condition, print -1 instead.

Note that the values of the subsequence must be printed, and not its positions. If two or more lexicographically equal subsequences occur in the vector in different positions such that they all satisfy the condition, you must print the values only once.

Examples

standard input	standard output
5 60522 14575 36426 79445 48772	14575
5 90081 33447 90629 3497 47202	3497

Note

It is guaranteed that the visible test cases are the only test cases generated without randomizing the size N and the vector V.

In the first example test case, the lexicographically smallest subsequence of the vector is [14575] and its hash is 405057492. Its complement is the subsequence $[60522, 36426, 79445, 48772]$, which has a hash of 980706017. As the lexicographically smallest subsequence already satisfies the condition, it should be printed.

Problem J. Just Not RU...

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	256 megabytes

Arthur has night classes at UnB every day, so he has to have dinner before going to class. He knows that most people usually have dinner at the RU, and he has nothing against the RU, but he thinks life is too short to miss out on different gastronomic experiences... However, not all of his friends share this point of view, so it is necessary to convince them.

Arthur and his friends have already not eaten at the RU on day 1, because "on the first day it is fine". But now his friends are determined to go to the RU in the next days. To prevent this, Arthur had an idea. He will propose the following: "we can go to the RU every day, just not on days that are multiples of X", such that X is a positive integer larger than 1. This seems reasonable to his friends at first, but little do they know that Arthur will choose several values of X so that they never go to the RU from day 1 to day N...

But Arthur must be careful so that his friends do not realize what he is doing! For this, he must say the smallest possible number of values X — otherwise, it would be too suspicious and he would be dragged to the RU!

Help him, and compute the smallest possible number of values X that Arthur must tell his friends so that they end up not going to the RU on any of the days from 1 to N.

Input

The input consists of a single integer N: the number of days ($1 \le N \le 10^6$).

Output

Print, on a single line, the smallest possible number of values X that Arthur must tell his friends so that they end up not going to the RU on any of the days from 1 to N.

Examples

standard input	standard output
1	0
4	2
12	5
1000000	78498

Note

In the first test case, since $N = 1$ and Arthur didn't go to the university restaurant on the first day, it's not necessary to tell any value for X.

In the second test case, Arthur can use $X = 2$ and $X = 3$ to not go on any of the $N = 4$ days (note that Arthur won't go on day 4, because 4 is a multiple of 2).

Problem K. Kalculating Damage

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 256 megabytes

Gus and Caio Fleury are two experienced Pokémon trainers. They are preparing for the next edition of the UnBalloon Pokémon League and, for that, they are running computerized battle simulations.

The simulation software they chose computes the damage of a single attack made by one Pokémon against another. To do this, it allows choosing the type of the attacking Pokémon, T_a, the power of the attack used, P, the type of the defending Pokémon, T_d, and the defending Pokémon's hit points, H. After this attack, the Pokémon's hit points are reduced to $max(0, H - P \cdot M(T_a, T_d))$, where M is the type damage modifier function. If the hit points are reduced to 0, the Pokémon is knocked out.

Gus and Caio Fleury prefer to use the most classic types: fire (F), grass (G), and water (W). For these specific types, the function M is defined as follows:

- If $(T_a, T_d) = (F, G)$, (G, W), or (W, F), then $M(T_a, T_d) = 2$.
- Otherwise, $M = \frac{1}{2}$.

Caio Fleury was using the simulator extensively to decide which strategy to use in the next battle. Meanwhile, Gus was thinking about contributing to the simulator with a custom message. Denoting by X the defending Pokémon's hit points after the attack, he thought of printing "Nocaute!" if $X = 0$, and "Sobraram X pontos de vida!" otherwise.

However, Caio Fleury has just challenged Gus to a battle, and now there is no time to implement this message. So now it is up to you to implement it!

Input

The first line of the input contains a single uppercase letter representing T_a, which can only be F, G, or W.

The second line also contains a single uppercase letter representing T_d, which can also only be F, G, or W.

The third line contains a single integer representing P ($2 \le P \le 10^6$). It is guaranteed that P is even.

The fourth line also contains a single integer representing H ($1 \le H \le 10^6$).

Output

Print, on a single line, the message proposed by Gus, according to the statement.

Examples

standard input	standard output
G W 6 10	Nocaute!
G W 4 10	Sobraram 2 pontos de vida!
G G 2 2	Sobraram 1 pontos de vida!

Problem L. Lady Gaga and the Echoing Coefficient

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Lady Gaga is a complete artist: unmatched in performances, extremely versatile, and full of diverse artistic references. Throughout her career, she has established herself as one of the most essential figures in the history of pop music, and as a voice for many who cannot find their own. To the delight of the *Little Monsters*, she constantly reinvents herself, always leaving her legion of fans eager for her next steps.

To further improve her songwriting skills, Lady Gaga studied a property of song lyrics called the Echoing Coefficient, which is simply the sum of the values of all Echoing Segments of a lyric. An Echoing Segment is a passage of a lyric that can be represented as the same passage repeated twice, and its value is the product of the number of lines that compose it and the total number of characters in those lines.

As an example, the following excerpt from Bad Romance is an Echoing Segment (because it can be represented as the repetition of the first three lines):

Oh-oh-oh-oh-oh

Oh-oh-oh-oh-oh-oh-oh

Caught in a bad romance

Oh-oh-oh-oh-oh

Oh-oh-oh-oh-oh-oh-oh

Caught in a bad romance

Formally, the lyrics of a song can be represented as a list L of lines, where the i-th line can be represented as a string L_i. A passage of a lyric is a contiguous sublist of L — if it is composed of all lines from the i-th to the j-th (with $j \geq i$), it is denoted by L_i^j. Thus, an Echoing Segment of L is a passage L_i^j such that there exists some list of lines V for which $L_i^j = V + V$, where $+$ represents the list concatenation operation. Therefore, the value of an Echoing Segment L_i^j is given by $(j - i + 1) \cdot \sum_{k=i}^{j} |L_k|$, where $|L_k|$ is the length of the string representing the k-th line. Finally, the Echoing Coefficient of a lyric is the sum of the values of all Echoing Segments of that lyric.

In her studies, Lady Gaga wants to analyze song lyrics and their respective Echoing Coefficients. To make this process more convenient, she asked her team for a program that receives a simplified song lyric (without uppercase letters, punctuation, or special characters) and prints the Echoing Coefficient of that lyric.

Help *Mother Monster* with a program that performs this calculation efficiently!

Input

The first line of the input contains an integer N: the number of lines in the song lyric ($1 \leq N \leq 5 \cdot 10^4$).

Each of the next N lines contains a string L_i, representing the i-th line of the lyric ($1 \leq |L_i| \leq 25$). It is guaranteed that L_i contains only lowercase characters from a to z.

Output

Print, on a single line, a single value: the Echoing Coefficient of the song lyric represented by the input.

Examples

standard input	standard output
14 ohohohohoh ohohohohohohoh caughtinabadromance ohohohohoh ohohohohohohoh caughtinabadromance raraahahah romaromama gagaohlala wantyourbadromance raraahahah romaromama gagaohlala wantyourbadromance	1284
12 cantreadmycantreadmy nohecantreadmypokerface shesgotmelikenobody cantreadmycantreadmy nohecantreadmypokerface shesgotmelikenobody cantreadmycantreadmy nohecantreadmypokerface shesgotmelikenobody cantreadmycantreadmy nohecantreadmypokerface shesgotmelikenobody	8184
2 a aaa	0
4 a b a b	16

Note

Note that the definition of an Echoing Segment refers to the concatenation of lists, not to the concatenation of all strings in the lists. Therefore, in the third sample test case, there are no Echoing Segments: even though "a" + "aaa" = "aaaa" and "aaaa" can be represented as "aa" + "aa", the list {"a", "aaa"} cannot be represented as the concatenation of two equal lists.

Problem M. Minimum Parking Cost

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2.5 seconds
Memory limit:	256 megabytes

Henrique wants to park his car in the parking lot of the University of Brasília (UnB).

The parking spots are numbered from 1 to n. Spot number i is characterized by two numbers: its distance d_i from the main entrance and the probability p_i that it is free. Henrique visits the spots sequentially, starting from spot 1. Upon arriving at spot i, two situations may occur:

1. With probability p_i, spot i is free. In this case, Henrique has two options: park in spot i (Henrique is extremely fast, so we may consider that he takes 0 seconds to park), or move on to the next spot, which is spot $i+1$ if $i \neq n$ and spot 1 if $i = n$.
2. With probability $1 - p_i$, spot i is occupied. In this case, Henrique is forced to continue to the next spot.

Each time Henrique moves on to the next spot, he spends 1 second. After visiting spot i once, it is possible that in subsequent visits its state will be different, since the UnB parking lot is very busy and a driver may have occupied or vacated the spot during the time Henrique took to return to it. This behaviour will be dictated by the probability p_i, which remains fixed throughout the entire problem.

Henrique wants to act so that the expected value of the sum of the total time spent choosing a spot and the distance of the spot where he finally parks is as small as possible. If Henrique acts optimally, what is this value?

Input

The first line of the input contains an integer n ($2 \leq n \leq 3 \cdot 10^5$).

The second line of the input contains n integers $d_1, d_2, \ldots, d_n$ ($1 \leq d_i \leq 10^6$).

Finally, the third line of the input contains n real numbers $p_1, p_2, \ldots, p_n$ ($0.001 \leq p_i \leq 1$). Each p_i is given with at most 3 decimal places.

Output

Print a single real number: the expected value of the sum of the total time spent choosing a spot and the distance of the spot where he parks under the best strategy. Your answer will be considered correct if the absolute or relative error does not exceed 10^{-6}. That is, if your answer is a and the correct answer is b, your answer will be accepted if and only if $\frac{|a-b|}{\max(1,|b|)} \leq 10^{-6}$.

Examples

standard input	standard output
7 8 9 9 4 6 3 2 1 1 1 1 1 1 1	7
2 1 1000000 0.5 1	3
6 31 415 92 6535 89 79323 0.461 0.884 0.500 0.972 0.149 0.111	38.015184381778709621 36

Note

In the first test case, Henrique should park in spot number 4. He takes a total of 3 seconds to reach it, and it is at a distance of 4 meters from the entrance, so the sum of the time spent and the distance is $3 + 4 = 7$. Since all spots are always free, he always achieves this value.

Problem N. Not Afraid to Fall

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

Luisa likes to practice on the UnBalloonforces website, where she solves problems and participates in contests. She follows the philosophy "Not Afraid to Fall", which says that one should participate in every contest, even if there is a risk of falling in title (equivalent to color on Codeforces), because that is the best way to train both skill and mindset. This worked so well that today she has the title of **Super Programmer**, which is the second highest on UnBalloonforces.

On UnBalloonforces, every problem has an integer rating between 1 and M. As a form of training for Luisa, her coach, Ruan, prepared a list of N problems with various ratings. He also sent a voice message saying: "Look, actually, just do the ones with rating greater than or equal to..." (Luisa could not hear the rating specified by Ruan).

Luisa has already solved all the problems, and now she is bored. Remembering the message, she wondered: "If the rating chosen by Ruan were i, what would be the sum of the ratings of the problems I should solve?" Help satisfy Luisa's curiosity by calculating this sum for every i from 1 to M!

Input

The first line of the input contains two integers: N and M, the number of problems in Luisa's list and the maximum rating of a problem on UnBalloonForces ($1 \le N, M \le 5 \cdot 10^5$).

The second line of the input contains N integers r_i: the i-th of them represents the rating of the i-th problem in the list ($1 \le r_i \le M$).

Output

Print, on a single line, M integers: the i-th of them must be the sum of the ratings of the problems Luisa would have to solve if Ruan had specified rating i.

Examples

standard input	standard output
5 5 2 4 2 4 2	14 14 8 8 0
3 2 1 1 2	4 2
8 9 5 9 1 2 3 4 7 6	37 36 34 31 27 22 16 9 9

Note

In the first test case, if the rating specified by Ruan had been 1 or 2, Luisa would need to solve all the problems, resulting in a sum of ratings equal to 14. If the specified rating were 3 or 4, Luisa would not need to solve the problems with rating 2, only the two with rating 4, and the sum would be only 8. Finally, if the specified rating were 5, Luisa would not need to solve any problem (sum 0).