

Problem A. Black Friday

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 1024 megabytes

After buying a monitor and a suitcase, then breaking the monitor by putting it in the suitcase on a flight, Leo resolved to make better personal finance decisions. However, he was challenged in this resolve when Black Friday came to pass. He ended up adding too many things to his shopping cart, to the dismay of his bank account. When all seemed lost, Leo remembered something. He had a coupon book lying around that he could use. These coupons were special and worked by first lowering his current shopping cart balance by a_i , and then giving a $d_i\%$ discount (removing $d_i\%$ of the adjusted balance). Due to a software error, Leo found that he could apply multiple coupons to the same shopping cart, and that applying all of the coupons in different orders could result in different final balances. Help Leo determine an optimal way to order the coupons that minimizes the final balance.

Input

The first line of input will consist of a single integer n ($1 \leq n \leq 10^5$) — the number of coupons Leo has.

The following n lines will consist of two real numbers a_i and d_i ($0.0 \leq a_i \leq 10^9$, $0.0 \leq d_i < 100.0$) — the amount to reduce the price by and the discount percentage of the i -th coupon.

Both values a_i, d_i will be printed to two decimal places. Also assume that Leo's shopping cart balance is large enough that it will never go negative after applying all of the coupons in any order.

Output

Output a line containing a permutation c of the numbers from 1 to n — the order that the coupons should be applied in. c_1 should correspond to the first coupon used, c_2 should correspond to the second coupon used, and so on. c should have n elements and be separated by spaces. There may be multiple optimal orderings, and any optimal ordering will be accepted.

Examples

standard input	standard output
2 16.65 0.00 0.00 16.65	2 1
3 200.00 10.00 10.00 20.00 50.00 5.00	2 3 1

This page is intentionally left blank

Problem B. Collecting Strawberries

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Madeline is stuck in a grid, where she can only move up, down, left and right by one cell. She is currently in the first cell of the last row of the grid and wants to collect a strawberry in the last cell of the first row. Initially, Madeline is extremely fast so it takes her zero seconds to travel to a new cell, however, part of her (often referred to as Badeline) wants to slow her down. For every pair of adjacent cells, Badeline can increase the time it takes Madeline to traverse between two cells by a positive real number, x , for a cost of $x \cdot c$. Note that c can differ between different pairs of cells. Assuming Madeline will take the shortest path to the strawberry after Badeline adds in the delays, determine the maximum shortest path Badeline can create for Madeline such that Badeline's total cost is at most k .

Input

The first line of input will consist of three integers, n , m , and k ($2 \leq n, m \leq 1000$, $1 \leq k \leq 10^9$) — the number of rows and columns in the grid and the max total cost.

For the following n lines, the i -th line will consist of $m - 1$ integers, $c_{r,i,j}$ indicating the cost rate of increasing the time it takes to traverse between cells (i, j) and $(i, j + 1)$, where (i, j) denotes the j -th cell of the i -th row.

For the following $n - 1$ lines, the i -th line will consist of m integers, $c_{c,i,j}$ indicating the cost rate of increasing the time it takes to traverse between cells (i, j) and $(i + 1, j)$, where (i, j) denotes the j -th cell of the i -th row.

All cost rates are between 1 and 10^9 .

Output

Output a single real number — the longest shortest path Badeline can create for Madeline. Your answer will be accepted if it has an absolute or relative error of 10^{-6} with the judge's solution.

Examples

standard input	standard output
2 2 3 1 1 1 1	1.5000000000
3 3 4 3 6 2 5 1 4 1 3 5 2 4 6	1.3333333333

This page is intentionally left blank

Problem C. File Your Taxes

Input file: `standard input`
Output file: `standard output`
Time limit: 2 seconds
Memory limit: 1024 megabytes

Tax day is coming! Quinn is very worried about his taxes but does not want to go to a tax lawyer to file them. He's decided to navigate the world of taxes on his own to find the very best way to file his taxes. The country Quinn lives in has two tax systems in which you can report income to be taxed. You cannot report a total income higher than m dollars.

A tax system is defined as follows:

- A tax system consists of n tax brackets.
- The tax system is defined by two **strictly increasing** sequences b and t of sizes $n + 1$ and n , respectively.
- b represents the tax bracket boundaries while t represents the tax rates of the brackets. Specifically, the i -th tax bracket is defined by three numbers b_i, b_{i+1}, t_i , defining an interval $[b_i, b_{i+1}]$ that is taxed at a rate of $t_i\%$.
- All of the tax brackets are disjoint and union to the full interval $[0, m]$. Therefore, $b_n = m$.

Evaluating how much tax you owe in a particular system depends on s , the starting point, and u , the income you report in that system. Then the income is represented by the interval $[s, s + u]$. For each bracket in the system, calculate the length of its intersection with the income interval and multiply it by the bracket's tax rate to get the tax paid in the bracket. Adding these taxes together yields the total tax paid in the system.

To actually report a total income of x , Quinn must first split it into y and z — the income he will report to tax system 1 and 2, respectively. Then the starting point for tax system 1 is 0 and the starting point for tax system 2 is y . Help Quinn determine the minimum amount of taxes he must pay if he splits x optimally. y and z must be non-negative, real values that sum to x .

However, Quinn does not yet know how much income he will earn this year, so he wants you to help him figure this out for multiple total incomes.

Input

The first line of input will consist of two integers n and m ($2 \leq n \leq 10^5, 1 \leq m \leq 10^9$) — the number of tax brackets both tax systems have and the maximum total income that can be reported, respectively.

Following that are the descriptions of tax system 1 and 2, respectively. The input formats for both are identical and as follows:

The first line consists of a strictly increasing array b of $n + 1$ real numbers ($0.0 \leq b_i \leq m, b_0 = 0.0, b_n = m$), representing the tax boundaries. Each value will have exactly two decimal places. The second line consists of a strictly increasing array t of n real numbers ($0.0 \leq t_i \leq 100.0$), representing the tax rates. Each value will have exactly five decimal places.

The next line of input will consist of a single integer q ($1 \leq q \leq 10^5$) — the number of queries. The last q lines of input will each consist of a single real number ($0.0 \leq x_i \leq m$), representing a query for a total income. Each value will have exactly two decimal places.

Output

Output q lines, each containing a single real number — the minimum tax paid for the respective query. Your answer will be accepted if it has an absolute or relative error of 10^{-6} with the judge's solution.

Example

standard input	standard output
3 10	0.33000000
0.00 2.00 3.00 10.00	1.20000000
20.00000 30.00000 100.00000	0.00000000
0.00 1.00 4.00 10.00	0.30000000
0.00000 50.00000 60.00000	0.00000000
12	4.80000000
1.66	4.20000000
4.00	0.33400000
1.00	0.40000000
1.60	0.43000000
0.00	0.70000000
10.00	0.38000000
9.00	
1.67	
2.00	
2.10	
3.00	
1.90	

Note

Tax System 1 is analogous to the ordinary income tax system in the US and Tax System 2 is analogous to the capital gains tax system in the US. This is, of course, a simplification of the US tax system. For example, you usually cannot choose how much of your income to distribute to each system. Do **not** use this material as financial, tax, or legal advice. For guidance on actual tax matters, consult a qualified tax professional or refer to official government resources. Also, do not forget to file your taxes.

Problem D. Hurricane Mapping

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 1024 megabytes

This is a run-twice interactive problem.

There is a hurricane brewing in the middle of the Atlantic Ocean and you have some sensors to deploy to track data about the storm, such as wind speeds and trajectory. Over the span of a one-kilometer square, you have placed n sensors so far, such that each side of the square has at least one sensor on it. The resulting area you can then scan with the sensors is the convex hull of the sensors.

You have one last sensor to place, then you must report the area that you can scan. However, the hurricane is quickly approaching, and with the strength of the hurricane, you know as soon as you get the last sensor in place, you will lose a significant amount of connection to the sensors.

Due to poor planning, you did not write down any of the coordinates of the sensors, and now only have access to one satellite that can communicate with the sensors. The satellite can ping a location within or on a five-kilometer square, however you do not know exactly where the original square is located, only that it is fully within or on the boundaries of the new square. The result of a ping will tell you whether that location is within or on the convex hull of the sensors. Using at most 10000 pings, you must determine the area of sensors' convex hull.

Interaction Protocol

This problem will be modeled as a run-twice interactive problem.

First Run

The first line of input consists of the word **ADD**.

The second line of input consists of a single integer, n ($1 \leq n \leq 1000$) — the number of initial sensors placed.

The following n lines consist of two integers, x_i and y_i ($0 \leq x_i, y_i \leq 1000$) — the point corresponding to the location of the i -th sensor. It is guaranteed that no two given points will be equal, and that each side of the bounding square will contain at least one point.

You must print two integers x_a and y_a ($0 \leq x_a, y_a \leq 1000$) — the point corresponding to the location of the added sensor.

Second Run

The first line of input consists of the word **AREA**.

You can make at most 10^4 queries by printing **QUERY** x y ($0 \leq x, y \leq 5000$), where x and y are integers. The interactor will reply with **INSIDE** if the point is inside (or on) the convex hull of all $n + 1$ sensors, and **OUTSIDE** otherwise. It is guaranteed that the coordinates of the sensors are integers in the range 0 to 5000, inclusive, and that their relative positions to each other were not altered. That is, for all i and j , the vector spanning point i to point j is equivalent in both coordinate systems.

Once you have determined the area of the sensors' convex hull, output a single line consisting of **AREA** A , where A is an integer corresponding to the **twice** the area of the convex hull. It can be proven that doubling the area will always result in an integer value.

If at any point in the second run:

- your program outputs an incorrect value of A ;
- your program outputs a line that doesn't match a format outlined above;
- your program uses more than 10000 queries;

- your program outputs a coordinate outside of the 5000 by 5000 bounding box;

the interactor will output “ERROR” on a new line. In that case, your program must immediately terminate without erroring, and you will receive a verdict of “Wrong Answer”.

The interactor is NOT adaptive. That is, the location of the initial bounding box will be determined before the second pass, and at no point will its location change regardless of what your program does.

Examples

standard input	standard output
ADD 4 0 500 1000 500 500 0 500 1000	500 500
AREA OUTSIDE INSIDE	QUERY 0 0 QUERY 2500 2500 AREA 1000000

Problem E. IPv6 Address Compression

Input file: standard input
Output file: standard output
Time limit: 5 seconds
Memory limit: 1024 megabytes

Sachin was investigating IPv6 addresses and the way they were compressed, and this gave him an idea for a new compression scheme. First, you have an address of n hexadecimal characters. Hexadecimal characters are the digits $0 - 9$ and the letters $a - f$. They are separated into blocks of k hexadecimal characters with colons in between blocks. To compress the string, perform the following process:

- Pick a hexadecimal character c and remove all occurrences of c in the string.
- Remove as few colons as possible so that there are no more than 2 colons in a row.

Sachin wants to know for a given string the smallest compression length. He also wants to see how changing the string affects the compression length, so he will give you multiple queries. In the i -th query, he will change all characters in the range $[l_i, r_i]$ in the string to the character c_i . After each query, report the smallest compression length.

Input

The first line of input consists of two integers n and k ($1 \leq k \leq n \leq 10^5$) — the length of the string and the block size, respectively. It is guaranteed that n is a multiple of k .

The second line of input consists of a string of n characters, where each character is a hexadecimal character ($0 - 9, a - f$).

The third line of input consists of a single integer q ($1 \leq q \leq 10^5$) — the number of queries.

The next q lines of input each consist of two integers l_i and r_i ($1 \leq l_i \leq r_i \leq n$), as well as a hexadecimal character c_i .

Output

Output q lines, where the i -th line consists of a single integer representing the smallest compression length of the string after processing the first i queries.

Example

standard input	standard output
8 2	7
0a80f2c3	8
5	8
2 3 0	7
4 6 e	4
8 8 c	
2 5 5	
2 7 9	

Note

After the first query in the example, the uncompressed string is `00:00:f2:c3`. The optimal character to remove is '0', which results in `::f2:c3`. Since there are no more than 2 colons in a row, no colons can be removed and the answer is 7.

After the fifth query in the example, the uncompressed string is `09:99:99:9c`. The optimal character to remove is '9', which results in `0:::c`. One colon needs to be removed, resulting in `0::c` with a length of 4.

This page is intentionally left blank

Problem F. K-Star Sets

Input file: **standard input**
 Output file: **standard output**
 Time limit: 3 seconds
 Memory limit: 1024 megabytes

Let $k \geq 2$. Define a k -star as a tree with $k + 1$ nodes, such that there is one central node connected to exactly k leaves. A k -star set is a non-empty graph where each connected component of that graph is a k -star. For a given tree, count the number of subsets of edges, such that the graph composed of just those edges and nodes that are endpoints of those edges forms a k -star set.

Input

The first line of input consists of two integers n and k ($2 \leq k < n \leq 2 \cdot 10^5$) — the number of nodes in the given tree and the size of each target star.

The following $n - 1$ lines each consist of two integers u and v ($1 \leq u, v \leq n$), denoting that there is an edge between nodes u and v .

Output

Output a single integer denoting the number of k -star sets of the given tree. Since this number can be large, output the answer under modulo 998244353.

Examples

standard input	standard output
5 3 1 5 2 5 3 5 4 5	4
7 2 1 2 1 3 2 4 3 5 3 6 3 7	11

This page is intentionally left blank

Problem G. LCM Subset Count

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Determine how many non-empty subsets of the numbers from 1 to n there are such that the least common multiple (LCM) of the numbers in the subset is a multiple of k .

Input

The first line of input will consist of a single integer t ($1 \leq t \leq 100$) — the number of test cases.

Each of the next t lines will consist of two integers n and k ($1 \leq n, k \leq 10^{18}$).

It is also guaranteed that all n and k can be factorized with primes smaller than 10^6 .

Output

Since the output can be large, print it modulo 998244353.

Example

standard input	standard output
4	8
4 4	44
6 6	114304
17 10	0
6 7	

This page is intentionally left blank

Problem H. New Password

Input file: **standard input**
Output file: **standard output**
Time limit: 3 seconds
Memory limit: 1024 megabytes

One day, Sachin's password p was found in a leaked passwords database. Since Sachin refuses to remember more than 1 password, this means he now has to change his password on every single account he has. Instead of coming up with a new password, he decides to make his new password derivable from his old one with the following transformation:

- Add every substring of p to a list. If string s is a substring of p multiple times, then it will appear in the list multiple times.
- Sort the list lexicographically.
- Concatenate the list into a new string p' .
- Use the L -th to R -th characters of p' as the new password.

Determine what Sachin's new password should be.

Input

The first line of input consists of a single integer n ($1 \leq n \leq 10^6$) — the length of Sachin's password p .

The second line of input consists of two integers L and R ($1 \leq L \leq R \leq \binom{n+2}{3}$) — the substring of p' to use as the new password.

The third line of input consists of a single string of n lowercase English letters — Sachin's old password p . It is guaranteed that $R - L + 1 \leq 10^6$.

Output

Output a single string of $R - L + 1$ characters — Sachin's new password.

Examples

standard input	standard output
3 1 10 aba	aaabababba
3 2 6 aba	aabab
7 5 17 bbbcaaa	aaaaaabbabbbb

This page is intentionally left blank

Problem I. Pirates on a Boat

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

*Catch me on a boat with a whole bunch of
pirates
It's a ****ing boat, and it's full of pirates*

—Yuno Miles, *Pirates on a Boat 2*

Captain Oliver is plotting a new expedition for his pirate crew. His navigation skills aren't the strongest, so he decided to randomly choose a point on the shore and a direction. He is now curious about his probability of reaching the opposite shore across the sea. He has now asked for your help to determine this probability by modeling it as a geometry problem!

He has modeled each shore as a polyline in the Cartesian plane, each starting at $x = 0$ and ending at the same value of x . Since the shores are separated by a sea, it is guaranteed that the maximum y value of the starting shore is strictly less than the minimum y of the ending shore. In addition, there are no bays; therefore, it is guaranteed that for each polyline, the x values of the points are strictly increasing.

Captain Oliver will then pick a point p uniformly randomly on the starting polyline. Let $\vec{n}$ be the normal vector to the segment that p is on, pointing towards the sea (i.e., it has a positive y component). Relative to $\vec{n}$, he will then pick an angle, θ ($-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}$), with probability $\frac{\cos(\theta)}{2}$. The boat's trajectory is then the ray starting at p and traveling at angle θ , in respect to $\vec{n}$. A trajectory reaches the opposite shore if it intersects the ending polyline and only intersects the starting polyline at p .

Input

The first line contains two integers, n and m ($2 \leq n, m \leq 10^6$) — the number of points on the starting shore's polyline, and the number of points on the ending shore's polyline.

The following n lines will each contain two integers, $x_{s,i}$ and $y_{s,i}$ ($0 \leq x_{s,i}, y_{s,i} \leq 10^9$) — the i -th point on the starting shore's polyline. It is guaranteed that $x_{s,i} < x_{s,i+1}$ for all $i \in [1, n-1]$.

The following m lines will each contain two integers, $x_{e,i}$ and $y_{e,i}$ ($0 \leq x_{e,i}, y_{e,i} \leq 10^9$) — the i -th point on the ending shore's polyline. It is guaranteed that $x_{e,i} < x_{e,i+1}$ for all $i \in [1, m-1]$.

It is guaranteed that the maximum value of $y_{s,i}$ is strictly less than the minimum value of $y_{e,i}$ and that $x_{s,1} = x_{e,1} = 0$ and $x_{s,n} = x_{e,m}$.

Output

Output a single real number — the probability that the trajectory successfully reaches the ending shore. Your answer will be accepted if it has an absolute or relative error of 10^{-6} with the judge's solution.

Examples

standard input	standard output
3 3 0 0 1 3 2 0 0 4 1 5 2 4	0.0911512657
5 7 0 0 2 2 3 1 4 2 6 0 0 6 1 3 2 4 3 7 4 4 5 3 6 6	0.3715825446

Problem J. Return of Puzzle Boy

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 1024 megabytes

Puzzle Boy is back for more! Ben's favorite hobby is puzzles (which is why he's referred to as Puzzle Boy). After he easily solved Nic's previous puzzle, Nic wants to give him a different kind of challenge. Nic has devised a word game played with a starting dictionary. In this game, Ben and Nic take turns writing new words in the dictionary, with Ben starting first. The person who can't think of a word to add to the dictionary on their turn loses the game.

The original set of rules for words in the dictionary is as follows:

- All of the words in the dictionary must be m letters long.
- Each word will only use the first c letters of the English alphabet.
- The words in the dictionary may not be distinct.

A newly added word w also has to follow these additional rules:

- The new word should be related to some word already in the dictionary. Therefore, there must exist some index i and word d that is already in the dictionary such that $w_i = d_i$.
- The new word must be novel and unique. Therefore, there must exist some index i where the letter w_i is not used at index i for any other word already in the dictionary.

Ben and Nic played a few rounds of the game with different starting dictionaries. At first, Ben would lose most of the time. But after saying "I understand it now," Ben started winning the majority of the games. To test whether Puzzle Boy truly solved the puzzle, Nic has challenged him to determine the number of different words he can add in the first turn of the game and win the game. Assume both Ben and Nic play optimally.

Input

The first line of input will consist of three integers n , m ($2 \leq n, m \leq 10^3$), and c ($2 \leq c \leq 26$) — the number of words in the dictionary, the length of each word, and the number of letters in the English alphabet that can be used.

The following n lines will each consist of a single string of m lowercase letters using only the first c letters of the English alphabet.

It is guaranteed that Ben can add a word to the dictionary at the beginning of the game.

Output

Output a single integer: the number of starting words that Ben can add to the dictionary on his first turn and win the game. Since this number can be large, output it modulo 998244353.

Examples

standard input	standard output
2 3 2 aaa aba	2
2 2 2 aa aa	0

This page is intentionally left blank

Problem K. Trapping Rain Water

Input file: **standard input**
Output file: **standard output**
Time limit: **1 second**
Memory limit: **1024 megabytes**

How hard can a LeetCode problem be?

— Some poor soul

Tired of the long and complex problem statements of ICPC contests, Samarth has decided to go back to practicing LeetCode. As a LeetCode Guardian, he's able to solve most LeetCode problems and doesn't find them difficult at all, especially since the problem statements tend to be short. He opened up the classic LeetCode problem, Trapping Rain Water. In this problem, you are given a 2D structure and have to compute where water is trapped in it after it rains from above. The 2D structure is given as an elevation map where each column has a certain height (see Sample 1 for a clear example).

However, Samarth is rusty when it comes to reading. He's never been particularly good at it and always had an ICPC teammate read problems for him. So when Samarth read the one sentence problem statement in Trapping Rain Water, he did not realize that the 2D structure is an elevation map. Instead, Samarth has the following assumptions about the 2D structure:

- The 2D structure is given as a grid of cells denoted by 'x' and '.', where '.' represents an empty cell and 'x' represents a block in a cell.
- The bottom row of the 2D structure is a fully occupied row of blocks.

Samarth also has the following assumptions about how water works:

- Water can only flow through empty cells (specifically, it can't flow through the space between adjacent blocks or between diagonally adjacent blocks).
- Water doesn't stick to blocks; if water runs down the side of a block, it won't stick to the bottom of the block.
- Water doesn't spread while falling. It falls straight downward due to gravity.
- Water that exits the grid will never enter the grid again.
- Air bubbles will not form when water flows into a trapped space.
- It rains for an arbitrarily long amount of time, and then the water will drain for an arbitrarily long amount of time.
- The siphon effect does not apply.

Help Samarth determine which cells will trap rain water. After all, how hard can a LeetCode problem be?

Input

The first line of input consists of two integers n and m ($3 \leq n, m \leq 10^6, n \cdot m \leq 10^6$) — the dimensions of the grid.

The following n lines of input each contain a row of the grid, represented as a string of m characters which are either '.' or 'x'.

It is guaranteed that the last row of the grid only has 'x'.

Output

Output the grid after it has rained. Specifically, output n lines of m characters each. Each character in the grid should be 'x' (block), '.' (empty cell), or '~' (filled with water).

Examples

standard input	standard output
<pre> 5 12x.... ...x...xx.x. .x.xx.xxxxxx xxxxxxxxxxxx </pre>	<pre>x.... ...x~~~xx~x. .x~xx~xxxxxx xxxxxxxxxxxx </pre>
<pre> 7 5 x...x x...x xx.xx x.x.x x...x xxxxx </pre>	<pre> x~~~x x~~~x xx~xx x.x.x x...x xxxxx </pre>
<pre> 8 7 xxx.xxxxx.x.. ...xx.. xx.x... x..x..x xxxxxxx </pre>	<pre> xxx.xxxxx~x.. ...xx.. xx~x... x~~x~~x xxxxxxx </pre>
<pre> 7 7 xxx.xxx x.xxx.. x.x.x.x x....x xxxxxxx </pre>	<pre> xxx.xxx x.xxx.. x~x~x~x x~~~~~x xxxxxxx </pre>
<pre> 5 7 xxxxx.xx.. x.x.x.x x.x...x xxxxxxx </pre>	<pre> xxxxx.xx.. x~x~x~x x~x~~~x xxxxxxx </pre>

Problem L. Trimming Hedges

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 1024 megabytes

Sahur has taken up gardening, and it's time for his annual checkup on one of the hedges he is maintaining. The hedge is represented as a series of segments of greenery of varying heights, represented by an array of n integers. To clean up the hedge, he needs to trim it so that every part of the hedge is the same height (he is fine with any height; he just wants it to be uniform).

To do this, he can use some of the strange-looking razors that he bought at a yard sale. Each razor is much longer than the hedge and has cutting blades that are one segment wide and evenly spaced apart on the razor. For every positive integer k , he has a razor with k segments of free space between each blade on the razor. In one application of a razor, he can line up the blades with some segments of the hedge, lift the razor to a certain height, and cut the hedge. A cut is applied simultaneously to all hedge segments that line up with the razor blades. If the hedge segment is taller than the blade, it will get cut down to the blade's height. Otherwise, the segment will not be cut. Since Sahur is lazy, he wants to minimize the number of times he has to use a razor to cut the hedge to his specifications.

Input

The first line of input consists of one integer n ($1 \leq n \leq 10^6$), the size of the array.

The second line of input consists of n integers $h_1, h_2, \dots, h_n$ ($1 \leq h_i \leq 10^6$) — the heights of the segments of the hedge.

Output

Output a single integer — the minimum number of times Sahur needs to use a razor to cut the hedge.

Examples

standard input	standard output
5 1 1 1 1 1	0
3 5 6 7	2

This page is intentionally left blank