

Problem A. Array Queries

Input file: **standard input**
Output file: **standard output**
Time limit: 4 seconds
Memory limit: 512 megabytes

Given a positive integer array A of length N . Your task is to perform the following operations:

- 1 l r a b c : for all indices i between l and r , inclusively, replace A_i with $A_i + a \cdot i^2 + b \cdot i + c$.
- 2 l r : find the minimum value of A_i , where i is between l and r , inclusively.

Input

The first line of the input contains two integers N and Q — the length of the array and the number of the queries, respectively ($1 \leq N, Q \leq 10^5$).

The second line of the input contains N integers A_1, \dots, A_N — the initial values of the array A ($0 \leq A_i \leq 10^{13}$).

Each of the following Q lines contains one query. The query of type 1 contains six integers $1, l, r, a, b$ and c , the query of type 2 contains three integers $2, l, r$ ($1 \leq l \leq r \leq N, 0 \leq a \leq 10^3, 0 \leq b \leq 10^8, 0 \leq c \leq 10^{13}$).

Output

For each query of type 2 print one integer — the answer on that query.

Example

standard input	standard output
6 7	10
1 1 4 2 4 0	25
1 1 6 0 2 0	82
1 3 3 1 0 1	
2 3 5	
1 1 5 0 3 3	
2 4 4	
1 2 6 3 2 1	
2 4 5	

Problem B. Bridge Rulebook

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

In the bridge, possible misconducts of the game table are regulated in accordance with the Code. The Code is a complex document in which references to other pages are found on each page...

The editorial board of the bridge magazine “Middle Slam” decided to release a new edition of the Code. The edition is a book of N pages, possessing the following properties:

- Each page contains references to one or several pages. All references go to pages with numbers greater than the current one.
- From the first page, you can get to any page through a chain of links.
- From any page, you can get to the N -th page through a chain of links.

How many ways are there to arrange the links on the pages of the book in this way? Since the answer can be very large, output it modulo 998 244 353.

Input

The input contains one integer N ($2 \leq N \leq 500$) — the number of pages in the edition.

Output

Output one integer — the remainder of the number of ways to arrange the links in an N -page book divided by 998 244 353.

Examples

standard input	standard output
3	2
5	122

Problem C. Collisions

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 512 megabytes

Given an ideal rubber ball and a track on which it rolls. We will consider the track as a straight line, where the ball starts at position 0, moving with speed 1 to the right, towards positive positions (so, after one second it will be at position 1, after two at position 2 and so on).

There are N walls installed, i -th wall is installed at the position X_i and have the toughness T_i . When the ball hits the wall, it bounces off and starts moving in the opposite direction. As the ball is ideal, it will continue to move at the same speed 1. After the collision, the toughness of the wall decreases by 1, and if it decreased to 0, that wall disappears.

Write the program that, for given positions and initial toughness of the walls, will determine how much time will pass from the start of the ball's movement to the last collision with the wall. Because the answer may be too big, print it modulo 998 244 353.

Input

The first line of the input contains one integer N ($1 \leq N \leq 50\,000$) — the number of the walls.

Each of the following N rows contains two integers X_i and T_i ($-10^9 \leq X_i \leq 10^9$, $1 \leq T_i \leq 10^9$) — position of the i -th wall and the initial value of its toughness, respectively.

You may assume that all X_i are pairwise distinct, that there are no wall at position 0, and that the ball is guaranteed to hit at least one wall.

Output

Print one integer — the amount of time in seconds from the start of the movement to the last collision with the wall modulo 998 244 353.

Examples

standard input	standard output
3 2 7 -2 1 -4 1	22
2 10000000 10836006 -10000000 87654321	535438694

Problem D. Distinct Roman Reorderings

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 512 megabytes

Given the string S , consisting of letters 'I', 'V', 'X', 'L', 'C', 'D' and 'M'. Your task is to count the Roman anagrams of the string, i.e. the number of different ways to reorder the letters in the string in a way that the resulting string is correctly written Roman numerals (degenerated reordering is counted as well).

Two reorderings are considered distinct, if the corresponding strings are not equal.

The following table from Wikipedia displays how the Roman Numerals are written:

Individual decimal places	Thousands	Hundreds	Tens	Units
1	M	C	X	I
2	MM	CC	XX	II
3	MMM	CCC	XXX	III
4		CD	XL	IV
5		D	L	V
6		DC	LX	VI
7		DCC	LXX	VII
8		DCCC	LXXX	VIII
9		CM	XC	IX

Note that:

- The numerals for 4, 9, 40, 90, 400 and 900 are written using “subtractive notation” where the first symbol is subtracted from the larger one (for example, for 40 (“XL”) ‘X’ (10) is subtracted from ‘L’ (50)). These are **the only** subtractive forms in standard use.
- A number containing several decimal digits is built by appending the Roman numeral equivalent for each, from highest to lowest.
- Any missing place (represented by a zero in the place-value equivalent) is omitted.
- The largest number that can be represented in the Roman notation is 3,999 (MMMCMXCIX).

Input

The first line of the input contains one integer T — the number of the test cases ($1 \leq T \leq 77\,777$). Each of the following T lines contains one test case — the string s , consisting of letters 'I', 'V', 'X', 'L', 'C', 'D' and 'M' ($1 \leq |s| \leq 15$).

Output

For each test case, print one integer — the number of the Roman anagrams of the given string.

Example

standard input	standard output
2	1
IC	1
XV	

Problem E. Editing The Graphics

Input file: standard input
Output file: standard output
Time limit: 4 seconds
Memory limit: 512 megabytes

In the new graphical editor you are provided with the image — a set of two or more triangles on the plane. The triangles in the image are satisfying the following conditions.

- These triangles never overlap each other (i.e for any two triangles the area of their intersection is equal to zero).
- Each triangle is connected to at least one other triangle by sharing a portion of its edge of non-zero length. If two triangles have only one common point, they are **not** considered connected.
- The union of all the given triangles is a simple polygon P , and the interior of that polygon is **completely** filled with triangles.

Find the coordinates of all the vertices of the polygon P .

Input

The first line of the input contains one integer N — the number of the triangles ($2 \leq N \leq 10^5$).

Each of the following N lines contains the coordinates of the vertices of one triangle $x_1, y_1, x_2, y_2, x_3, y_3$ ($-10^9 \leq x_i, y_i \leq 10^9$), given in counterclockwise order.

You may assume that the given set of triangles meet the requirements from the problem statement.

Output

Consider the result being a simple polygon P with M vertices. Then print M lines, each line containing two integers — x - and y -coordinates of the vertex.

In the first line, print the vertex with smallest y -coordinate, in case of the tie choose one with smallest x -coordinate.

In the second line print the vertex, that is adjacent to the first one and is next after it when the vertices are listed counterclockwise from the positive direction of the x -axis centered on the first printed vertex. The subsequent vertices are defined as the vertex that was not printed and is adjacent to the previously printed vertex.

Examples

standard input	standard output
2 1 1 5 5 1 5 2 2 4 2 4 4	1 1 2 2 4 2 4 4 5 5 1 5
3 0 1 2 1 1 3 2 1 3 3 1 3 2 1 4 1 3 3	0 1 4 1 3 3 1 3

Problem F. Friendly Figures

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

The robot is drawing the picture on the paper.

The robot draws a picture in the following way:

- Draws one line segment per one step.
- If the new line segment have the common point with any of existing segments, this point is the endpoint for both segments.

Consider two line segments as the parts of the same *figure*, if the segments are connected directly or indirectly, i.e. there exists the sequence of segments with one of those segments as first element, another one as last, and the property that any two neighboring segments in the sequence share the endpoint.

So after each step, the paper contains several figures.

Consider the figure *friendly*, if it can be drawn with a single stroke, i.e. one can draw complete figure without removing the writing instrument from the paper even once and without tracing the same line twice. However, it **does not matter** how many times the same end point is passed.

Your task is to print number of the friendly figures on the paper after each drawing step of the robot.

Input

The first line of the input contains one integer N — the number of the steps ($1 \leq N \leq 2 \cdot 10^5$).

i -th of the following N lines describe i -th step and contains four integers $x_{i,1}, y_{i,1}, x_{i,2}, y_{i,2}$, in order — the coordinates of the endpoints of the segment drawn by robot on the i -th step ($0 \leq x_{i,1}, y_{i,1}, x_{i,2}, y_{i,2} \leq 10^9$, the length of the segment is strictly greater than zero).

You may assume that for any two segments their intersection is either empty or is exactly one point, that is endpoint for both segments.

Output

Print N lines. In i -th line print one integer — the number of the friendly figures on the paper after step i .

Example

standard input	standard output
8	1
5 1 8 4	2
5 7 2 4	3
10 6 10 2	2
2 4 5 1	2
5 1 5 7	2
8 4 5 7	0
8 4 10 2	1
10 6 8 4	

Problem G. Guess The Integer

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 512 megabytes

This is an interactive problem

Your task is to guess an integer N between 1 and 10^6 , secretly written by the jury program. To do that, you may operate with the special register X . At the beginning of the game $X = N$.

You may ask the queries in the form `? d`, where d is the integer between 0 and 10^6 . The jury program **replaces** current value of X with the $X + d$. If $X + d > 2^{20}$, you lose. Otherwise the jury program answers 1, if $X + d$ is the square of some integer, and 0 if there is no such integer. You can ask no more than 2023 queries.

If you decide that you know enough to tell the value of N , you tell the answer in the form `! N`. If it is correct, the solution is accepted. This action is **not** counted as a query.

You may assume that the interactor is **not** adaptive, i.e. value of N does not changes at the time of interaction.

Interaction Protocol

The interaction is started by your program to printing the query. The query have the format `? d` ($0 \leq d \leq 10^6$). The jury program answers 0 or 1 depending of new value of the register X (1 if for some integer q $q \cdot q = X$, and 0 otherwise). Then you ask the next query and so on. If you want to print the answer, use format `! N`.

Do not forget to flush the output after each query (or after printing the answer), otherwise you can get the Idleness Limit error.

Example

standard input	standard output
1	? 2
0	? 8
1	? 8
1	? 75
	! 7

Problem H. Hits of Eighty-Bits

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

The famous star of rap Long Double, known also as Eighty-Bits, have written a new track.

The only thing left before he can go into the studio and record the new track is to combine the music with the lyrics. There shall be intro, the song itself and the final part.

The intro and the final part may be arbitrary (ever empty!), but the main part (song itself) shall consist of two or more lines, each line **of the same length**. So Eighty-Bits task is to divide the lyric into those parts, and then split the song itself to the lines of the same length

Eighty-Bits asked the HitGPT bot for the advice. The HitGPT answered that the song part of the text must be divided into lines in the way that maximizes the *rhyme value* of the text.

HitGPT defined the rhyme value as follows: let N be the number of lines in the song part (without the intro and final part) and S be the length of the longest common suffix that all those lines have in common. Then the rhyme value is equal to $N \cdot S$.

Given the lyrics of the track, help Eighty-Bits to find the largest possible rhyme value.

Input

The first and only line of the input consists of a string, the lyrics of the track. The string contains only lowercase letters (a-z) and underscores (_). Its length is between 2 and 15 000. Note that the underscore **does not** have any special meaning and shall be treated as plain letter.

Output

Print one integer — the large possible rhyme value.

Examples

standard input	standard output
buck_buck_buck_buck_buck_buck_buck	35
qqqqqqqqqqqqqqqqqqqqqq	20

Note

In the Sample 1, the lines can be formed in the following way:

```
buck (intro)
_buck
_buck
_buck
_buck
_buck
_buck
```

The final in this track is empty. That gives $N = 7$ and $S = 5$, so the value is 35.

Problem I. Incredible Quick Sort

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

Bytica invented the new way to sort the integers, and she called it the XORting.

Given a list of integers a_1, a_2, \dots, a_N , she can quickly find an integer X so that

$$a_1 \oplus X \leq a_2 \oplus X \leq \dots \leq a_N \oplus X,$$

where \oplus means bitwise XOR, so all she needs to do then is replace a_i with $a_i \oplus x$ and voila, the list is sorted!

Your task is the implement multiple XORting queries.

You are given a list of non-negative integers a_1, \dots, a_N and Q queries of the form p_i, v_i , which means that the number a_{p_i} changes to v_i in i -th query.

Your task is to find $Q + 1$ integers c_0, c_1, \dots, c_Q , where c_i is the **smallest non-negative** integer X such that the list $a_1 \oplus X, \dots, a_N \oplus X$ is sorted, after the first i changes have been performed. If there is no such X , print -1 instead of that value of X .

Input

The first line of the input contains an integer N ($1 \leq N \leq 10^6$) the number of elements of the array a .

The second line contains N integers a_1, a_2, \dots, a_N ($0 \leq a_i < 2^{30}$), the initial values of the elements of the array.

The third line contains an integer Q ($0 \leq Q \leq 10^6$), the number of queries.

The following Q lines contain two integers p_i ($1 \leq p_i \leq N$), and v_i ($0 \leq v_i < 2^{30}$) each, which means that the number a_{p_i} changes to v_i .

Output

Print $Q + 1$ lines, i -th line containing one integer c_{i-1} .

The number c_i should be the smallest possible integer X that XORts the list after the changes $1, 2, \dots, i$ have been performed (or -1 if there is no such X).

Example

standard input	standard output
3	0
0 1 4	2
3	-1
2 7	4
3 3	
1 4	

Problem J. Jams

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

An application is being developed that allows drivers to choose the optimal speed to avoid traffic jams.

In the process of working on this application, data collection on the operation of traffic lights was carried out as follows.

Suppose there is a traffic light installed at the intersection that works according to the following rules:

if at time t the traffic light changes from red to green, then at time $t + g$ the traffic light will switch from green to yellow (that is, the green light will be on from time t to time $t + g$, excluding time $t + g$), at time $t + g + y$ it will switch to red (that is, the yellow light will be on from time $t + g$ to time $t + g + y$, excluding the last one), and at time $t + g + y + r$ it will switch from red to green (that is, the red light will be on from time $t + g + y$ to time $t + g + y + r$, excluding the last one).

The signal color sensor installed in the car records the time at which the car crosses the intersection and the color of the traffic light at that moment.

Based on the collected data, your task is to determine the probability that at the given time T the traffic light will have the specified color c .

Input

The first line of the input contains three integers g , y , and r - the duration of the green, yellow, and red signals, respectively ($0 \leq g, y, r \leq 10^8$).

The second line of the input contains a single integer n - the number of records collected by the sensor ($1 \leq n \leq 1000$).

Each of the following n lines contains one record. The record consists of two fields: the time t ($0 \leq t \leq 10^9$) and a single letter that specifies the color of the traffic light ('g' for green, 'y' for yellow, and 'r' for red).

The last line contains, in a similar format, the time T and the color of the traffic light c for which a request is made.

Output

Output a single number - the probability that at time T the traffic light will have the color c with an absolute or relative error of 10^{-3} .

Examples

standard input	standard output
4 4 4 3 2 g 18 y 34 r 5 g	0.2500000000000000
4 4 4 4 2 g 6 y 10 r 14 g 4 r	0.0000000000000000
6 6 6 6 5 g 6 g 9 y 12 y 15 r 19 r 7 g	1.0000000000000000

Problem K. Keyboards Production

Input file: **standard input**
Output file: **standard output**
Time limit: 4 seconds
Memory limit: 512 megabytes

The Thinkbox company plans to build the new factory in jungle for their new product — the wireless keyboard with the pointstick, illuminated keys and half-of-year duration per one charge. The success of this project will guarantee the Thinkbox long time domination on the market.

The factory will be placed under the forest and can be shaped as a square. The forest can be represented as $R \times C$ matrix, each cell representing the type of trees that is dominating in that cell. The tree types are denoted by lowercase English letters. The factory terrain then shall be the square $F \times F$ cells, where $1 \leq F \leq \min(R, C)$, that is aligned with the cells of the forest.

The CEO of the Thinkbox wants the factory to be as secure as it possible. Nowadays is too easy for other big hardware companies to take the picture of the forest, where the factory shall be placed, using the drones. So CEO of Thinkbox wants to choose the tree types placement in the square such that there will be at least K squares with the same side in the forest that have exactly the same distribution of tree types (i.e. for any two of K places there exists an parallel shift that turns one of those squares into another considering the tree types). Note that those squares may overlap.

The CEO of the Thinkbox wants factory to be as cheap as possible. The tax for the factory does not depend on the length size, so the higher value of F means less relative price per cell.

Given the map of the forest and the value of K , find the maximal length of side F for the underforest factory, or print -1 , if the CEO's requirement can not be held at all.

Input

The first line of the input contains three integers R , C , and K ($1 \leq R, C \leq 600$, $2 \leq K \leq R \cdot C$) — the number of rows and columns in the matrix representing the forest, and the number of the similar squares, respectively.

Then R lines follow, each consisting of C lowercase English letters — the map of the forest. The cells with the same dominating type of the trees are denoted with the same letters, with distinct dominating types — with the distinct letters.

Output

Print one integer — the maximal side of the square F . If it is impossible to find K squares with the same distribution of the tree types, print -1 instead.

Examples

standard input	standard output
4 5 2 baacd bbaab bbbaa bbbbbb	3
3 3 5 cbb dxb xxx	-1

Note

In the Sample 1, the squares 3×3 with the upper left corner (0,0) and the upper left corner (1,1) have the same shape

baa
bba
bbb