Problem A. Mad Matrix Multiplication

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

In mathematics, matrix multiplication is a binary operation that produces a matrix from two matrices. Matrix multiplication is a basic tool of linear algebra, and as such has numerous applications in many areas of mathematics, as well as in applied mathematics, statistics, physics, economics, and engineering. That is said by Wikipedia and that is true. Also in computer science matrix multiplication is widely used in machine learning, computer graphics, cryptography and other areas. In neural networks, calculating matrix multiplication usually takes at least 90 percent of the total computation time, so the performance of a lot of the fancy machine learning-based things of our time depends on the efficiency of the matrix multiplication implementation.

That was some useful information for you, but absolutely useless in the legend of this task, because the task is very simple. You are given two matrices, multiply them.

Input

First line contains three integer numbers N, M, K. $(1 \le N, M, K \le 1000)$. The values of the matrices A and B are given in the next lines.

Matrix A is of the size $N \times M$ and is described in N lines with M integer numbers in each.

Matrix B is of the size $M \times K$ and is described in M lines with K integer numbers in each.

Matrix descriptions are separated with empty lines.

The values of matrix elements are limited by following inequations: $0 \le A_{ij} < 10^9+7$ and $0 \le B_{ij} < 10^9+7$

Output

Print matrix $C = AB$ of the size $N \times K$ in the N lines with K integer numbers in each. Print elements of the matrix C modulo $10^9 + 7$.

By the definition of matrix multiplication j-th printed number in the i-th row should be equal $\left(\sum_{t=0}^{m-1} A_{it} \cdot B_{tj}\right) \ mod\ 10^9 + 7$

Example

standard input	standard output

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

Problem B. Don't Try To Hash It

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

Someone has sent you a string of N small Latin characters. This unknown man also asked you to perform some operations on this string. You, being a brave, clever and inquisitive person, agreed to do everything that you are asked to do. Of course, you will write a program that fulfills all the queries you are given without a shadow of a doubt!

The given queries are of two different kinds.

First type of queries asks to replace all the characters of some substring by the given character.

Second type of queries asks to compare two given substrings of the same length lexicographically and output -1 if the first substring is smaller, 1 if the second substring is smaller and 0 if substrings are equal.

Input

The first line contains single integer N - length of the given string s. $1 \leq N \leq 10^5$.

The second line contains one string of the length N: $s_1 s_2 s_3 ... s_N$.

The third line contains single integer Q - number of queries.

Each of the next Q lines contains description of each query in the following formats:

1 l r c - query of the first type, it asks you to replace all the characters from l-th to r-th with the given lowercase Latin letter c. $(1 \leq l \leq r \leq N)$

2 k a b - query of the second type. It asks you to compare substrings $s_{a..a+k-1}$ and $s_{b..b+k-1}$. $(1 \leq k \leq N,\ 1 \leq a, b \leq N - k + 1)$

Example

standard input	standard output
7 dababac 5 2 3 2 4 2 5 1 2 1 1 6 a 2 5 1 2 2 5 2 3	0 1 0 -1

Problem C. Integer Points Easy

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

There is a circle of radius R centered at the origin of coordinate plane.

Count the number of integer points inside or on the border of the circle.

That is the number of points with integer coordinates, which are located at a distance of no more than R from the center of the circle.

Input

Single integer R - radius of the circle. $1 \leq R \leq 10^6$

Output

Single integer - number of integer points in the circle.

Examples

standard input	standard output
1	5
3	29

Problem D. Integer Points

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

There is a circle of radius R centered at the origin of coordinate plane.

Count the number of integer points inside or on the border of the circle.

That is the number of points with integer coordinates, which are located at a distance of no more than R from the center of the circle.

Input

Single integer R - radius of the circle. $1 \leq R \leq 10^9$

Output

Single integer - number of integer points in the circle.

Examples

standard input	standard output
1	5
3	29

Problem E. Reachability

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

You are given an oriented graph of N vertices.

For each vertex v in the given graph you should compute R_v - number of vertices u reachable from vertex v.

Vertex u is considered to be reachable from vertex v if there exists a path of oriented edges which starts with v and ends with u.

Input

The first line contains two integers: number of vertices N and number of edges M. $1 \le N \le 5 \cdot 10^4,\ 1 \le M \le 2 \cdot 10^5$

Each of the next M lines contains description of one edge: two integers u and v, which means that there is an oriented edge from u to v. $1 \le u, v \le N$, $u \ne v$.

Output

Print N lines with a single integer on each. Integer on the i-th line should be equal to R_i - number of reachable vertices from vertex i.

Example

standard input	standard output

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

Problem F. Fibonacci Multiverse

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

Once Eugene decided that it would be nice to calculate all the Fibonacci numbers. He took the numbers 0 and 1 as the initial values and calculates one more Fibonacci number every second. Obviously, he will do this until he gets bored. You must agree that this is a very interesting activity and he will not get tired of it soon. Thus, in the first second of this activity he got the number 1, in the second - the number 2, then 3, 5, 8, 13, 21 and so on. But you know, because of all these time travels, jumps between universes, magic and other cool stuff, which we observe in different films and tv-series, the matter of space-time collapses and our global multiverse becomes very unstable. So collisions between universes sometimes happen, due to which the time continuum is disrupted and the timeline of our universe is exposed to factors from other universes. Surprisingly, when such things happen, Eugene's activity is also affected.

In more detail, when a collision occurs, it changes a certain period of time of the past, present and future, as a result of which the global knowledge of humanity in our universe changes and Eugene begins to use a different formula for calculating of new Fibonacci numbers. Namely, each collision of universes touches a certain period of time from L to R seconds from the beginning of Eugene's activity and forces him to use some another formula of kind $F_n = a \cdot F_{n-1} + b \cdot F_{n-2}$ for some integers a and b instead of familiar to us $F_n = F_{n-1} + F_{n-2}$.

You built a brilliant machine which allows you to observe all the universes collisions. And since you also love Fibonacci numbers a lot, you monitor the Eugene's activity and sometimes you want to know which Fibonacci number Eugene computes at the T-th second from the start of his activity with current state of the universe. So it's a good idea to write a program which will help you satisfy your curiosity.

Input

First line of the input contains a single integer N - number of lines in the log of your brilliant machine about the universes collisions and your curiosity since the beginning of time. Each next of N lines is one of the two kinds:

@ L R a b - line of the first type describes new collision which affects period of time form L-th to R-th second of Eugene's activity and forces him to use coefficients a and b for Fibonacci formula during that time. $1 \leq L \leq R \leq 10^9$ and $1 \leq a, b \leq 10^9$

? T - line of the second type describes the query of your curiosity about the moment T of the current state of the universe. $1 \leq T \leq 10^9$

Output

For each line of the second type in the input you should print single number on the separate line - the value of Fibonacci number computed by Eugene at the corresponding moment T. Since this number may be very big, print its value modulo $10^9 + 7$. Print answers on queries in the same order as they appear in the input.

Example

standard input	standard output
9 ? 1 ? 3 ? 5 ? 9 @ 1 3 2 1 ? 5 @ 3 8 3 5 ? 5 ? 9	1 3 8 55 29 425 38675

Note

Explanation of the Example.

Initial values which Eugene uses for Fibonacci numbers are 0 and 1.

Until first collision happens, Eugene uses simple formula $F_n = F_{n-1} + F_{n-2}$ during all the time, so the computed by him numbers from the 1st second of his activity are $1, 2, 3, 5, 8, 13, 21, 34, 55, 89...$ and answers on the first 4 queries are the numbers: $1, 3, 8, 55$.

After first collision, described by the line @ 1 3 2 1, Eugene uses formula $F_n = 2 \cdot F_{n-1} + F_{n-2}$ at the seconds $1, 2$ and 3, so computed by him numbers are $2, 5, 12, 17, 29, 46...$ and answer on the next query is equal to 29.

Problem G. A Lot Of Numbers Easy

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	18 megabytes

You are given an array of integers. The size of the array can be very large, but fortunately, most of the numbers in the array are boring for you, since they occur exactly 2 times. Unique numbers are much more interesting. You would like to explore and analyze the unique numbers of a given array, you may even make some puzzles with them, but first you need to find them. So let's do that!

Input

The first line contains a single integer N - size of the given array. $1 \le N \le 5 \cdot 10^6$.

The second line has N integer numbers $A_1, A_2, ..., A_N$ - elements of the array. $0 \le A_i \le 10^9$.

It is guaranteed that there are only K unique numbers in that array, and each other number, except that K, occurs exactly two times in that array. $1 \le K \le 9$.

There are two kinds of tests in this version of the task:

1) $1 \le N \le 10^6$ and $1 \le K \le 9$

2) $1 \le N \le 5 \cdot 10^6$ and $K = 1$

Output

On the first line print single integer K - number of unique numbers in the given array.

On the second line print unique numbers of the given array sorted in **ascending order**.

Examples

standard input	standard output

`5`
`4 7 7 3 4`
	`1`
	`3`

`4`
`0 74 47 0`
	`2`
	`47 74`

`11`
`7 13 5 41 13 41 25 7 2 2 125`
	`3`
	`5 25 125`

Note

Pay attention to the unusual memory limit in this task!

Problem H. A Lot Of Numbers

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	18 megabytes

You are given an array of integers. The size of the array can be very large, but fortunately, most of the numbers in the array are boring for you, since they occur exactly 2 times. Unique numbers are much more interesting. You would like to explore and analyze the unique numbers of a given array, you may even make some puzzles with them, but first you need to find them. So let's do it!

Input

The first line contains a single integer N - size of the given array. $1 \leq N \leq 5 \cdot 10^6$.

The second line has N integer numbers $A_1, A_2, ..., A_N$ - elements of the array. $0 \leq A_i \leq 10^9$.

It is guaranteed that there are only K unique numbers in that array, and each other number, except that K, occurs exactly two times in that array. $1 \leq K \leq 9$.

Output

On the first line print single integer K - number of unique numbers in the given array.

On the second line print unique numbers of the given array sorted in **ascending order**.

Examples

standard input	standard output
5 4 7 7 3 4	1 3
2 47 74	2 47 74
11 7 13 5 41 13 41 25 7 2 2 125	3 5 25 125
9 0 1 2 3 4 5 6 7 8	9 0 1 2 3 4 5 6 7 8

Note

Pay attention to the unusual memory limit in this task!

Problem I. Three Queries On The Tree

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

You have a rooted tree with the root in the vertex 1, where some numbers are written on the vertices.

You should perform queries of the 3 types:

1) Add some value to all the numbers on the given simple path in the tree.

2) Count the numbers which are less or equal to the given value on the given simple path.

3) Count the numbers which are less or equal to the given value in the given vertex subtree.

Input

The first line contains an integer N - number of vertices in the tree. $1 \leq N \leq 10^5$.

Each of the next $N-1$ lines contains two numbers u and v - some edge of the tree. It's guaranteed that given values describe correct tree.

The next line contains N numbers $A_1, A_2, ..., A_N$, where A_i - is the initial number written on the vertex i. $-10^8 \leq A_i \leq 10^8$

The next line contains single integer M - number of queries. $1 \leq M \leq 10^5$.

Each of the next M lines describes one query of the one of 3 types:

1 v u x - add value x to all the numbers on the simple path between vertices v and u. $1 \leq v, u \leq N$, $-10^3 \leq x \leq 10^3$.

2 v u x - count the numbers which are $\leq x$ on the simple path between vertices v and u. $1 \leq v, u \leq N$, $-10^9 \leq x \leq 10^9$.

3 v x - count the numbers which are $\leq x$ in the subtree of vertex v. $1 \leq v \leq N$, $-10^9 \leq x \leq 10^9$.

Output

For each query of the 2nd and 3rd type, in the same order as they are given in the input, print one number on the separate line - answer on the query.

Example

standard input	standard output

`5`
`1 2`
`2 3`
`3 4`
`2 5`
`2 -3 5 9 5`
`6`
`1 1 2 3`
`2 1 3 1`
`1 1 4 -4`
`2 4 5 0`
`1 5 3 2`
`3 2 4`
	`1`
	`1`
	`2`