Problem A. Autonomous Cities

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

There are N cities in Byteland, cities are connected by $N - 1$ bidirectional roads such as each city can be reached from the another one using one of several those roads. A city is called *autonomous*, if it have only one road leading to (and from) this city.

Given N, find the least possible number of autonomous cities.

Input

Input consists of one integer N ($2 \le N \le 10^9$) — number of cities in Byteland.

Output

Print one integer — least possible number of autonomous cities.

Examples

standard input	standard output
2	2
3	2

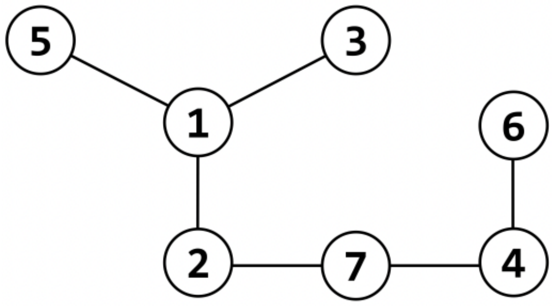
Problem B. Query on a Tree

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

You are given a tree where vertices are labeled with integers $1, 2, \ldots, N$.

For a subset of vertices $S \subseteq \{1, 2, \ldots, N\}$, we say two vertices (u, v) are *connected under* S if there exists a path that only passes through the vertices in S. Note that this includes endpoints of the path, so $u, v \in S$ should hold.

For example, consider the following tree and the set $S = \{1, 2, 3, 4, 5, 6\}$.



In this case, $(1, 2)$, $(3, 5)$ and $(4, 6)$ are connected under S, while $(1, 6)$ and $(2, 7)$ are not connected under S.

Let *strength*(S) be the number of pairs of vertices (u, v) such that $u \neq v$ and (u, v) are connected under S. You are given Q queries, where each query contains a set S. For each query, you should compute the quantity *strength*(S).

Input

The first line contains a single integer N, the number of vertices ($2 \leq N \leq 250\,000$).

Each of the next $N - 1$ lines contains two space-separated integers a and b: the vertices connected by an edge ($1 \leq a, b \leq N$). Together, the edges form a tree.

The next line contains a single integer Q, the number of queries ($1 \leq Q \leq 100\,000$).

Each of the next Q lines contains a query, denoted by space-separated integers. A query starts with an integer K, the size of the set ($1 \leq K \leq N$). It is followed by K distinct integers from 1 to N in arbitrary order: the vertices of set S.

The sum of K in each test case is at most $1\,000\,000$.

Output

For each of the Q queries, print a single line with the integer *strength*(S) as defined above.

Example

standard input	*standard output*
7 1 2 1 3 1 5 2 7 4 6 4 7 6 1 1 2 1 2 4 1 2 3 4 5 1 2 4 6 7 6 1 2 3 4 5 6 7 1 2 3 4 5 6 7	0 1 3 10 7 21

Problem C. Customity

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

Let's define *customity* of two adjacent elements a_i and a_{i+1} of the array A as $|a_i - a_{i+1}|$.

Given an integer array A. For one operation you may add to any element of the sequence arbitrary real number. Your task is to convert the array to one where maximum customity will be minimized.

Calculate minimal number of operations needed to do that.

Input

First line of the input contains one integer N — length of the given array ($2 \leq N \leq 10^5$). Second line contains N integers a_i ($-10^6 \leq a_i \leq 10^6$) — elements of the given array A.

Output

Print one integer — minimal number of operations needed to obtain the array where maximum customity is minimized.

Examples

standard input	standard output
2 1083 6006	1

Problem D. Building Bombing

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

KAIST has a series of N buildings in a row, numbered from 1 to N, from left to right. Building i has a height of h_i. Building i is visible from the left if and only if every building on its left has a height strictly less than h_i.

Your lab is located in building number L. Since your favorite number is K, you want to make your lab building the K-th tallest building visible from the left. To achieve your goal, you will blow up some of the buildings.

For example, suppose there are $N = 7$ buildings in a row and their heights are $[10, 30, 90, 40, 60, 60, 80]$. Your lab is located at building number $L = 2$ and your favorite number is $K = 3$. After blowing up buildings 3 and 7, the buildings visible from the left will be buildings 1, 2, 4, and 5. Then your lab becomes the 3rd tallest building visible from the left, as desired.

What is the minimum number of buildings to blow up to make your lab building the K-th tallest building visible from the left?

Input

The first line contains three space-separated integers N, L, and K.

The second line contains N space-separated integers $h_1, \ldots, h_N$.

- $1 \le L \le N \le 100\,000$
- $1 \le K \le 10$
- $1 \le h_i \le 10^9$ $(1 \le i \le N)$

Output

Output the minimum number of buildings to blow up to make your lab building the K-th tallest building visible from the left. If it is impossible to do so, output -1 instead.

Examples

standard input	*standard output*
7 2 3 10 30 90 40 60 60 80	2
3 2 2 30 20 10	-1

Problem E. Divide and Convert

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

Given a string, consisting of letters 'I', 'V', 'X', 'L', 'C', 'D' and 'M'. Your task is to split the string to the continuous substrings such as:

- Each substring is a valid Roman numeral.
- The sum of integers, represented by those numerals, is maximum possible.

Input

Input contains one non-empty string, consisting of letters 'I', 'V', 'X', 'L', 'C', 'D' and 'M'. Length of the string does not exceed 10^6.

Output

Print one integer — maximal sum of the integers in Roman system, achieved by splitting the string to the valid Roman numerals.

Examples

standard input	standard output
VII	7
ICL	151

Note

In the first sample we can take the number as it is. In the second sample the maximum can be reached, for example, by splitting the string by two substrings: 'I' (1) and 'CL' (150), what gives 151 as the sum.

Problem F. Making Number

Input file: `standard input`
Output file: `standard output`
Time limit: 1.5 seconds
Memory limit: 1024 mebibytes

You are given two positive integers X and Y of the same length in base 10. Let us define Z as the positive integer in base 10 satisfying the following conditions.

- The digits of Z should be a rearrangement of the digits of X. Leading zeros in Z are not allowed. For example, if $X = 1103$, Z can be 1103 or 3101, but Z cannot be 2110, 301, nor 0131.
- $Y \leq Z$.
- Z is the minimum value satisfying the above conditions.

You have to perform Q queries. Each query is one of the following:

- Given i and x, change the i-th digit of Y into x.
- Given i, output the i-th digit of Z. If there is no such Z, print -1.

The digits of an integer are numbered from left to right starting from 1. For example, The third digit of 1234 is 3.

Input

The first line contains two space-separated integers, X and Y.

The second line contains a single integer Q, the number of queries.

Each of the following Q lines contains space-separated integers describing the queries. Each line has one of the following forms, where the first integer represents the type of the query:

- "`1` i x": Change the i-th digit of Y to x.
- "`2` i": Output the i-th digit of Z. If there is no such Z, print -1.

It is guaranteed that there is at least one query of type `2`.

Let $\text{len}(A)$ be the number of digits in a positive integer A.

- $1 \leq X, Y < 10^{100\,000}$
- $1 \leq Q \leq 100\,000$
- $\text{len}(X) = \text{len}(Y)$
- The first digits of X and Y are not 0.
- For a query of type `1`, $1 \leq i \leq \text{len}(Y)$, $0 \leq x \leq 9$, and if $i = 1$, then $x \neq 0$.
- For a query of type `2`, $1 \leq i \leq \text{len}(Y)$.

Output

For each query of type `2`, output a single line with the answer to the query.

Examples

standard input	*standard output*
3304 1615 6 2 3 2 4 1 1 3 2 2 1 2 4 2 1	3 4 0 3
838046 780357 10 2 1 2 2 1 2 4 2 3 2 4 1 4 5 2 5 2 6 1 1 9 2 2	8 0 3 4 6 8 -1
2950 9052 4 2 1 2 2 2 3 2 4	9 0 5 2

Problem G. One Path

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

You are given a tree T consisting of N vertices. Each edge has a positive integer weight.

You can perform the following operation on the given tree.

- Delete an edge from the graph, then add a new edge between any two distinct vertices. The weight of the new edge must be the same as the weight of the deleted edge. The resulting graph need not be a tree.

We define the weight of a path as the sum of the weights of the edges on the path. The distance between two vertices u and v is defined as the weight of the *shortest path* from u to v — having the minimum weight. If there is no such path, we define the distance as 0.

The weight of a graph is the maximum of the weights between any two vertices.

Your task is to find the largest weight of the graph that can be obtained by performing the operation exactly i times, for $i = 0, 1, \ldots, K$.

Input

The first line contains two space-separated integers, N and K.

The i-th of the following $N - 1$ lines contains three space-separated integers, u_i, v_i, and w_i, representing an undirected edge that connects two different vertices u_i and v_i with a weight of w_i.

It is guaranteed that the edges form a tree.

- $2 \le N \le 2000$
- $0 \le K \le 2000$
- $1 \le u_i < v_i \le N$ $(1 \le i \le N - 1)$
- $1 \le w_i \le 10^9$ $(1 \le i \le N - 1)$

Output

Output $K+1$ space-separated integers. The i-th integer should be equal to the largest weight of the graph that can be obtained by performing the operation exactly $i - 1$ times.

Examples

standard input	*standard output*
5 1 1 3 2 4 5 4 3 4 3 2 3 7	14 16
7 2 1 2 4 2 3 6 2 4 2 4 5 5 2 6 1 4 7 3	13 20 21

Problem H. Permutation Arrangement

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

You are given an array a of length N. Each element of a is either -1 or an integer between 1 and N. Each number between 1 and N appears at most once in a. Also, no two adjacent elements of a differ by exactly 1.

You are to find the lexicographically smallest permutation p of $\{1, 2, \ldots, N\}$ satisfying the following.

- if $a_i \neq -1$, then $a_i = p_i$ $(1 \leq i \leq N)$;
- $|p_i - p_{i+1}| \neq 1$ $(1 \leq i \leq N-1)$.

Input

The first line contains one integer, N.

The second line contains space-separated N integers: elements of the array a.

- $1 \leq N \leq 200\,000$
- $1 \leq a_i \leq N$ or $a_i = -1$ $(1 \leq i \leq N)$
- $a_i \neq a_j$ or $a_i = -1$ $(1 \leq i < j \leq N)$
- $|a_i - a_{i+1}| \neq 1$ $(1 \leq i \leq N-1)$

Output

If there is no permutation p satisfying the condition, then output a single integer -1.

Otherwise, output the lexicographically smallest permutation p.

Examples

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

Problem I. Establish The Minimum

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

Given an number. Reorder the characters in its decimal representation in a way such as the resulting integer will be minimal possible.

Both given and resulting number shall conform the following rules to be valid:

- Decimal representation consists of digits between '0' and '9' inclusively and special characters '-' and '.', usage of those characters is described later. You may assume that representation contains atleast one digit.
- The number may be negative (then first character is representation is the '-') or non-negative (then no special character is used). There cannot be '-' character anywhere else than at first position.
- The number may be finite decimal fraction (in this case the representation contains exactly one dot '.' as decimal point; the dot can be placed only between two digits) or integer, in this case decimal representation does not contains '.'.
- The representation may start from zero (or have zero as second character after '-' sign) only if the number is equal to zero or if immediately after it goes the decimal point. However, zeroes at the end of the representation of fractional numbers are OK, for example, the representations -123.200, or 2.0000, or 0.0 are valid.
- If the absolute value of the given integer is equal to zero, the '-' sign cannot be used.
- Lentgh of the decimal representation is not greater than 10^4.

Input

Input file contains one non-empty string consisting of no more than 10^4 characters — valid decimal representation of the number.

Output

Print minimal possible valid number which can be obtained by rearranging characters in the given decimal representation.

Examples

standard input	standard output
123405	102345
-3.1416036	-6643311.0

Problem J. Squirrel Game

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

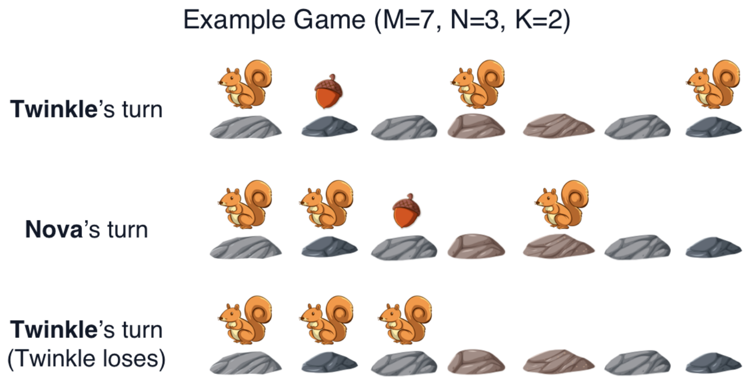
Twinkle and Nova are walking in a national park. There are M stones laid out in the park at positions $1, \ldots, M$, from left to right. There are also N squirrels on the stones at $x_1, \ldots, x_N$, from left to right. The squirrels are on different stones from each other, and they are all facing left.

Twinkle suggests the following game to Nova. Twinkle and Nova take turns alternately. On each turn, a player has to place an acorn on one of the stones without a squirrel. Also, there must be at least one squirrel to the right of the acorn.

After placing an acorn, the leftmost K squirrels among the squirrels to the right of the acorn start running towards the acorn at the same time. (If there are less than K squirrels to the right of the acorn, all of them start running.) All the squirrels run at the same speed. Once any of the squirrels reaches the acorn, all the squirrels immediately stop. The squirrel who has reached the acorn puts the acorn into its cheek pouch, effectively removing the acorn from the stone.

If there is no valid stone to place an acorn on, the player currently taking the turn immediately loses.

Twinkle goes first. Determine who will win if Twinkle and Nova both play optimally.



Input

The first line contains three space-separated integers, M, N, and K.

The second line contains N space-separated integers $x_1, \ldots, x_N$.

- $1 \leq N \leq M \leq 100\,000$
- $1 \leq K \leq 10$
- $1 \leq x_1 < \ldots < x_N \leq M$

Output

If Twinkle wins, output `Twinkle`. Otherwise, output `Nova`.

Examples

standard input	*standard output*
7 3 2 1 4 7	Nova
7 3 1 1 4 7	Twinkle

Problem K. Find Two Triangles

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

Find two triangles with integer lengths of sides, given perimeter P, one with maximal non-zero area, other with minimal non-zero area.

Input

Input contains one integer P — given value of the perimeter.

Output

In the first line print three integers — lengths of sides of the triangle with given perimeter and **maximal** area. In the second line print in same format lengths of sides of the triangle with given perimeter and **minimal non-zero** area. If there is more than one solution, print any of them.

If there are no non-degenerated triangles with integer lengths of sides and perimeter P, print -1 instead.

Examples

standard input	standard output
3	1 1 1 1 1 1
4	-1

Problem L. Village Planning

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	1024 mebibytes

As the mayor of the RUN town, you are planning to build a new village. The village consists of houses and bidirectional roads connecting two different houses. Roads are organized in such a way that no two roads connect the same pair of houses. In other words, the village can be treated as a simple graph where houses corresponds to vertices and roads corresponds to bidirectional edges. Note that the village may be disconnected.

You want your village to be as simple as possible. Therefore, for any distinct houses i and j, there should be at most K simple paths from house i to house j.

Let N be the number of houses. The score of the village is

$$\prod_{1 \le i < j \le N} A_{f(i,j)},$$

where $f(i,j)$ is the number of simple paths from house i to house j.

While the number of houses is not determined yet, you know that it will be an integer between 2 and M. You should calculate the sum of the scores for all possible villages with N houses for each N from 2 to M.

Since the answers can be large, output them modulo 998 244 353.

Input

The first line contains two space-separated integers M and K.

The second line contains $K+1$ space-separated integers $A_0, \ldots, A_K$.

- $2 \le M \le 100\,000$
- $0 \le K \le 3$
- $1 \le A_i < 998\,244\,353$ $(0 \le i \le K)$

Output

For each N from 2 to M, output the sum of the scores for all possible villages with N houses, modulo 998 244 353. The answers should be separated by single spaces. Note that $998\,244\,353 = 119 \cdot 2^{23} + 1$ is a prime number.

Examples

standard input	*standard output*
4 0 2	2 8 64
5 1 3 4	7 327 96721 169832849
6 2 5 6 7	11 1566 3000672 306031599 466869291
7 3 8 9 10 11	17 5427 31856976 326774674 449014006 997476587

Problem M. Window Arrangement

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

KAIST is running out of budget — they need some money! They thought the dormitories were way too luxurious compared to the other buildings of KAIST; they planned to sell all the dormitory buildings and build a new completely non-aesthetic one.

The new dormitory will have grid shape — it can't be more boring than this — of size $N \times M$, each cell being a room for the students to live in. We are going to add some windows, because we want students to get some sunlight during the daytime!

We plan to have exactly $w_{i,j}$ windows in the room (i, j). Each room is surrounded by 4 unit edges of the grid. A window can be built on the side of a unit edge of the grid, and at most one window can be built on each side of a unit edge — so each cell has between 0 and 4 windows. A window is one-sided: a window on the opposite side of a unit edge does not count as a window in the room.

Unfortunately, students will experience huge discomfort when their private space is watched by someone else through the window. The **total discomfort** is the number of pairs of students $\{a, b\}$ such that a and b can see each other's private space through the window.

In other words, if a unit edge has windows on both sides, the total discomfort increases by the product of the numbers of people living in the two rooms sharing the window.

You are given $w_{i,j}$, the number of windows planned for the room (i, j), and $p_{i,j}$, the number of people living in the room (i, j). Your task is to find the minimum total discomfort that can be achieved by arranging the windows properly.

Input

The first line contains two integers N and M: the dimensions of the grid.

Each of the following N lines contains M space-separated integers $p_{i,j}$.

Each of the following N lines contains M space-separated integers $w_{i,j}$.

- $1 \le N \le 50$
- $1 \le M \le 50$
- $1 \le p_{i,j} \le 1000$ $(1 \le i \le N, 1 \le j \le M)$
- $0 \le w_{i,j} \le 4$ $(1 \le i \le N, 1 \le j \le M)$

Output

Output one integer: the minimum total discomfort.

Examples

standard input	*standard output*
4 3 1 7 10 7 2 8 7 9 10 4 6 4 3 3 3 3 2 4 4 3 4 2 2 3	178
4 3 2 2 9 9 8 4 8 4 5 7 5 2 0 1 0 1 0 1 0 0 1 0 1 0	0