Problem A. Add on Prefix

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You have an array a.

In one operation you can take some prefix of the array and increase all elements of this prefix by 1, you should pay x $(x > 0)$ coins for it.

You may assume that you have an infinite number of coins, so you can perform any number of operations.

After you perform all operations, for each maximum in the array (for each element with the largest value) you will get y $(y > 0)$ coins.

Your total earning is equal to $-x \times$ (number of operations that you performed) $+\, y \times$ (number of maximums in the array after all operations).

Let's say that the maximum total earning is equal to $f(a, x, y)$.

You are given three arrays of positive integers $(b_1, \ldots, b_n)$, $(x_1, \ldots, x_n)$, $(y_1, \ldots, y_n)$.

For each i, such that $1 \le i \le n$, you need to find $f((b_1, \ldots, b_i), x_i, y_i)$.

Input

The first line of input contains one integer n $(1 \le n \le 300\,000)$: length of the given arrays.

Next n lines contain three integers each, i-th of them contain three integers b_i, x_i, y_i $(1 \le b_i, x_i, y_i \le 10^9)$.

Output

Print n integers, each on the new line. In i-th of them you should print $f((b_1, \ldots, b_i), x_i, y_i)$.

Examples

standard input	standard output

2
2 1 2
1 2 1
	2
	1

5
1 1 2
2 1 3
2 1 4
2 1 5
3 1 6
	2
	5
	11
	19
	28

Problem B. Best Partition

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

Let's call directed graph **impressive** if there exists at least one vertex, such that all other vertices are reachable from it. Note that the graph with one vertex is impressive.

You are given a weighted **impressive** graph.

You need to delete some edges of this graph and partite vertices of this graph into k non-empty groups, such that for each group, induced subgraph on its vertices is **impressive**, your goal is to minimize the weight of remaining edges.

Input

The first line of input contains two integers n, m, k ($1 \le n, m \le 50\,000, 1 \le k \le n$): the number of vertices and edges in the given graph.

Next m lines contain a description of edges, i-th of them contain three integers u, v, w ($1 \le u, v \le n; u \ne v$), ($1 \le w \le 20\,000$), describing an edge $u \to v$ with weight w.

It is guaranteed that the given digraph is **impressive**.

Output

Output one integer: smallest total weight of edges that you can leave in the graph, such that it would be possible to split all vertices into k non-empty **impressive** induced subgraphs.

Examples

standard input	standard output

3 2 1
1 2 1
2 3 2
	3

3 2 2
1 2 1
2 3 2
	1

3 2 3
1 2 1
2 3 2
	0

Problem C. Circles Covering

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given n circles on the plane.

You need to find a circle with the smallest radius which contains all given circles inside.

Input

The first line of input contains one integer n ($1 \le n \le 50$): the number of given circles.

The next n lines of input contain a description of circles, i-th of them contains three integers x_i, y_i, r_i ($-1000 \le x_i, y_i \le 1000, 1 \le r_i \le 1000$).

Output

Output one number: the smallest possible radius of the circle that contains all given circles. Your answer will be considered correct if its relative or absolute error doesn't exceed 10^{-4}.

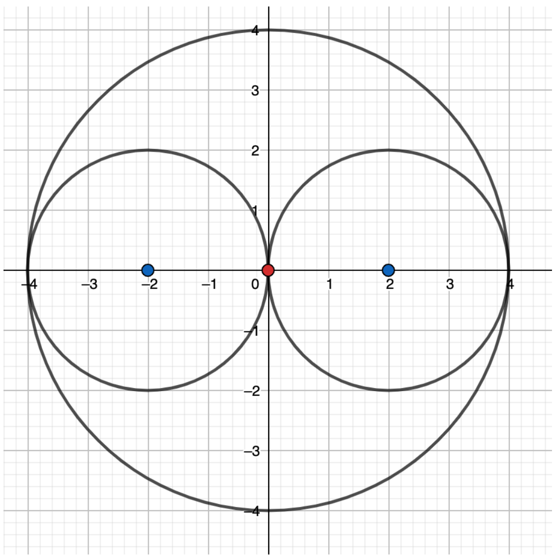
Examples

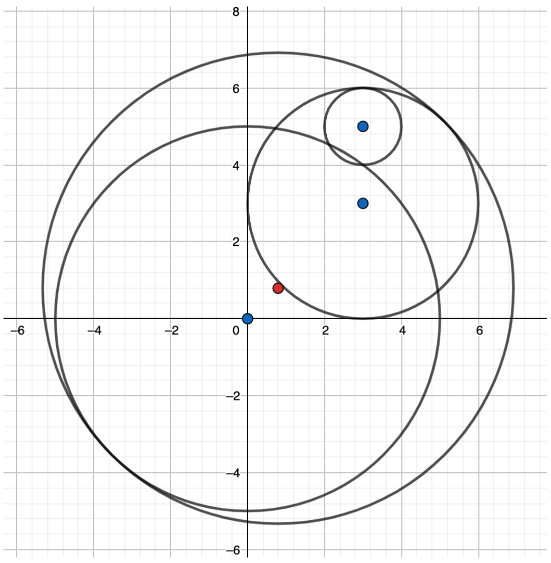
standard input	standard output

2
-2 0 2
2 0 2
	4

3
0 0 5
3 3 3
3 5 1
	6.1213203

Note





Problem D. Domino for Young

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given a Young diagram.

Given diagram is a histogram with n column of lengths $a_1, a_2, \ldots, a_n$ ($a_1 \geq a_2 \geq \ldots \geq a_n \geq 1$).

Your goal is to find the largest number of non-overlapping domino that you can draw inside of this diagram.

Input

The first line of input contain one integer n ($1 \leq n \leq 300\,000$): the number of columns in the given histogram.

The next line of input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 300\,000, a_i \geq a_{i+1}$): the lengths of columns.

Output

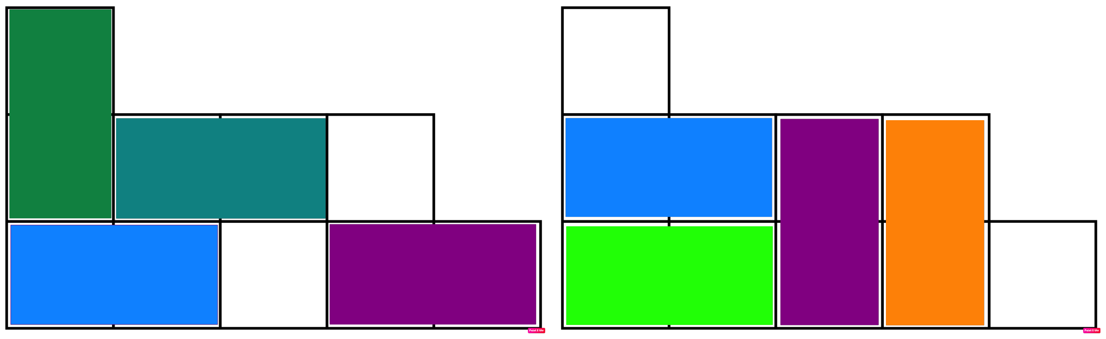
Output one integer: the largest number of non-overlapping domino that you can draw inside of the given Young diagram.

Example

standard input	standard output
5 3 2 2 2 1	4

Note

Some of the possible solutions for the example:



Problem E. Evil Segments

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given a sequence $a_1, a_2, \ldots, a_n$ of positive integers.

Let's call a subsegment $a_l, a_{l+1}, \ldots, a_r$ of this sequence **evil** if there exists some element i, such that $l \le i \le r$ and there are no j such that $j \ne i, l \le j \le r$ and $a_j = a_i$ (i.e subsegment is evil if and only if there exists some value x that present exactly once on this segment).

You need to find the number of ways to partite given sequence into evil subsegments, modulo $998\,244\,353$.

Input

The first line contains one integer n ($1 \le n \le 300\,000$): the number of elements in the sequence.

The next line contains n integers, $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$): given sequence.

Output

Print one integer, the number of ways to partite given sequence into evil subsegments, modulo $998\,244\,353$.

Examples

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

Problem F. Forbidden Triple

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given a set A of n different integers.

You need to find a subset S of A such that $|S| \geq \frac{n}{3}$ and there are no triple a, b, c such that $a \in S, b \in S, c \in S$ and $a + b = c$.

It can be proved that it is always possible.

Input

The first line of input contains one integer n ($3 \leq n \leq 300\,000$): the number of integers in A.

The second line of input contains n different integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^6$).

Output

In the first line of output print one integer m ($\frac{n}{3} \leq m \leq n$): the number of integers in S.

In the second line print m different integers $s_1, s_2, \ldots, s_m$ ($1 \leq s_i \leq 10^9, s_i \in A$).

There should be no triple i, j, k such that $1 \leq i, j, k \leq m$ and $s_i + s_j = s_k$.

If there are several possible solutions, you can print any.

Example

standard input	standard output
3 1 2 3	2 2 3

Problem G. Graph Subpaths

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given a DAG (Directed Acyclic Graph) on n vertices. For each edge $u \to v$ in this graph, u is strictly smaller than v.

Also, you are given k paths on this DAG.

For each i from 2 to n, you need to calculate the number of paths in DAG from 1 to i such that no given path is contained inside it as a substring. In other words, you need to calculate the number of paths from 1 to i such that each given path has at least one edge which **is not contained** on it.

As the answer may be very large, you only need to find it modulo 998 244 353.

Input

The first line of input contains two integers n, m ($3 \le n, m \le 100\,000$): the number of vertices and edges of the given graph.

Next m lines contain two integers each, i-th of them contain two integers a_i, b_i ($1 \le a_i < b_i \le n$) that describe an edge $a_i \to b_i$ in the DAG.

It is guaranteed that each pair of vertices connected with at most one edge.

The next line contains one integer k ($0 \le k \le 50\,000$): the number of given paths.

Each of the next k lines contains integer l ($2 \le l \le n$) and l vertices $v_1, v_2, \ldots, v_l$ ($1 \le v_i \le n$, all edges $v_i \to v_{i+1}$ are present in graph). It is guaranteed that total sum of l is at most 100 000.

Output

Output $n - 1$ integers: the number of paths in DAG from 1 to i such that no given path is contained inside it as a substring, modulo 998 244 353, for each i, such that $2 \le i \le n$.

Examples

standard input	standard output

3 3
1 2
2 3
1 3
0
	1 2

3 3
1 2
2 3
1 3
2
2 1 2
3 1 2 3
	0 1

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

Problem H. Happy Cactus

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given a cactus graph, in this graph each edge lies on at most one simple cycle.

It is given as m edges a_i, b_i, weight of i-th edge is i.

Let's call a path in cactus **increasing** if the weights of edges on this path are increasing.

Let's call a pair of vertices (u, v) **happy** if there exists an increasing path that starts in u and ends in v.

For each vertex u find the number of other vertices v, such that pair (u, v) is happy.

Input

The first line of input contains two integers n, m ($1 \le n, m \le 500\,000$): the number of vertices and edges in the given cactus.

The next m lines contain a description of cactus edges, i-th of them contain two integers a_i, b_i ($1 \le a_i, b_i \le n, a_i \ne b_i$).

It is guaranteed that there are no multiple edges and the graph is connected.

Output

Print n integers, required values for vertices $1, 2, \ldots, n$.

Examples

standard input	standard output

3 3
1 2
2 3
3 1
	2 2 2

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

Problem I. Invertation in Tournament

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given a tournament — complete directed graph.

In one operation you can pick any vertex v and change the direction of all edges with v on one of the ends (i.e all edges $u \to v$ change their orientation to $v \to u$ and vice versa).

You want to make a tournament strongly connected with the smallest possible number of such operations if it is possible.

Also, if it is possible, you need to find the number of ways to make these number of operations to make graph strongly connected (two ways are different if for some i vertex that we chose on i-th operation in one way is different from vertex that we chose on i-th operation in another way). This number may be very large, so you just need to find it modulo 998 244 353.

Input

The first line of input contains one integer n ($3 \le n \le 2000$): the number of vertices in the tournament.

Following n lines contain a description of the given tournament, each of them contains a binary string of length n. If j-th character of i-th string is equal to '1', then the graph has an edge $i \to j$.

It is guaranteed that there are no edges $i \to i$ and the graph has **exactly one** edge among $i \to j$ and $j \to i$ for different i and j.

Output

If it is not possible to convert tournament to strongly connected with the given operations, output "-1".

Otherwise, output two integers: the smallest number of operations that you need to make the given graph strongly connected and the number of ways to do this number of operations to make graph strongly connected, modulo 998 244 353.

Examples

standard input	standard output
3 010 001 100	0 1
4 0010 1000 0100 1110	-1
6 010000 001000 100000 111001 111100 111010	2 18

Problem J. Just Counting

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given a rooted tree of n vertices, numbered with integers from 1 to n, where 1 is the root of the tree.

Also, you are given a q vertical paths on this tree.

Your goal is to find the number of ways to color edges of the tree into two colors, such that each such path contains both colors.

Two ways are different if there is an edge, such that its color is different in these ways.

The answer may be very large, so output it modulo 228 228 239.

Input

The first line of the input contains one integer n ($2 \le n \le 2000$): the number of vertices in the tree.

The next line of the input contains $n-1$ space-separated integers, $p_2, p_3, \ldots, p_n$, ($1 \le p_i < i$), the parent of vertex i is p_i.

The next line contains one integer q ($1 \le q \le 2000$): the number of the given vertical paths.

Next q lines contain by two integers each, i-th of them contains two space-separated integers a_i and b_i $1 \le a_i < b_i \le n$, it is guaranteed that a_i is contained in the path from the root to b_i (in other words, a_i is the ancestor of b_i), and describes the given vertical path from a_i to b_i.

Output

Output one integer "— the number of ways to color edges of the tree into two colors, such that each given path contains both colors, modulo 228 228 239.

Examples

standard input	standard output

5
1 2 3 4
1
1 5
	14

7
1 1 1 2 3 4
3
1 5
1 6
1 7
	8

2
1
2
1 2
1 2
	0

Problem K. K Integers

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given a permutation $p_1, p_2, \ldots, p_n$.

In one move you can swap two adjacent values.

You want to perform a minimum number of moves, such that in the end there will exist a subsegment $1, 2, \ldots, k$, in other words in the end there should be an integer i, $1 \le i \le n - k$ such that $p_i = 1, p_{i+1} = 2, \ldots, p_{i+k-1} = k$.

Input

The first line of input contains two integers n, k ($1 \le n \le 200\,000, k \le n$): the number of elemengts in the permutation and k.

The next line of input contains n integers $p_1, p_2, \ldots, p_n$: given permutation.

Output

Print one integer, the minimum number of moves that you need to make a subsegment with values $1, 2, \ldots, k$ appear in the permutation.

Examples

standard input	standard output
5 4 5 4 3 2 1	6
3 3 1 2 3	0

Problem L. Long Beautiful Integer

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given an integer x of n digits $a_1, a_2, \ldots, a_n$.

Also, you are given a positive integer $k < n$.

Let's call integer $b_1, b_2, \ldots, b_m$ **beatiful** if $b_i = b_{i+k}$ for each $1 \le i \le m - k$.

You need to find the smallest **beautiful** integer $y \ge x$.

Input

The first line of input contains two integers n, k ($2 \le n \le 200\,000, 1 \le k < n$): the number of digits in x and k.

The next line of input contains n digits $a_1, a_2, \ldots, a_n$ ($a_1 \ne 0$, $0 \le a_i \le 9$): digits of x.

Output

In the first line print one integer m: the number of digits in y.

In the next line print m digits $b_1, b_2, \ldots, b_m$ ($b_1 \ne 0$, $0 \le b_i \le 9$): digits of y.

Examples

standard input	standard output
3 2 353	3 353
4 2 1234	4 1313

Problem M. Modulo Equality

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

You are given two arrays of non-negative integers, $a_1, a_2, \ldots, a_n$, $b_1, b_2, \ldots, b_n$.

You need to find the smallest non-negative integer x, such that it would be possible to find some permutation $p_1, p_2, \ldots, p_n$ such that $(a_i + x) \pmod{m} = b_{p_i} \pmod{m}$.

Input

The first line contains two integers n, m ($1 \le n \le 2000, 1 \le m \le 10^9$): number of elemens in arrays and m.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i < m$).

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i < m$).

It is guaranteed that there exists some non-negative integer x, such that it would be possible to find some permutation $p_1, p_2, \ldots, p_n$ such that $(a_i + x) \pmod{m} = b_{p_i} \pmod{m}$.

Output

Print one integer, the smallest non-negative integer x, such that it would be possible to find some permutation $p_1, p_2, \ldots, p_n$ such that $(a_i + x) \pmod{m} = b_{p_i} \pmod{m}$.

Examples

standard input	standard output

3 2
0 0 0
1 1 1
	1

5 10
0 0 0 1 2
2 1 0 0 0
	0