Problem A. Aho-Parasick

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds (8 seconds for Java)
Memory limit:	512 mebibytes

We didn't come up with 5/10 or better joke, but some word mangling should suffice.

The Aho-Corasick algorithm takes a set of strings, which we will denote **the dictionary** as the input. Then it constructs the following structure:

There is a vertex corresponding to every prefix of one or more strings from the dictionary. There is also a vertex corresponding to the empty string. Two trees are formed from this set of vertices.

The first one is called the **trie** and it consists of all edges between nodes corresponding to strings s and t, such that t can be obtained from s by appending a single character.

The second one is called the **suffix links** and it consists of all edges between nodes corresponding to different strings s and t, such that s is a suffix of t and there is no string w different from s and t with a corresponding trie node, such that s is a suffix of w and w is a suffix of t.

It can be shown that both the trie and the suffix links are trees. In this problem all edges are undirected.

In this problem trees are treated as sets of edges, and edges are treated as unordered pairs of vertices.

You are given two trees A and B on the same set of vertices S. Construct a dictionary, such that

1. Applying the Aho-Corasick algorithm to this dictionary yields trees isomorphic to the given ones. In other words, let V denote the set of nodes (vertices) of the constructed trie and suffix links, T — the constructed trie and L — the suffix links. There must exists a bijection $f : S \to V$, such that $\forall_{c1,c2 \in S}\ \{c1, c2\} \in A \iff \{f(c1), f(c2)\} \in T, \{c1, c2\} \in B \iff \{f(c1), f(c2)\} \in L$.
2. Total length of all strings in the dictionary doesn't exceed $3 \cdot 10^5$.

The alphabet size is limited by the total length of all strings. The answer is guaranteed to exist.

Note that there are pairs of trees such that it is possible to construct a dictionary which will satisfy the first requirement, but it is not possible to satisfy both. Such test cases are not present in this problem, however, because the answer is guaranteed to exist. In other words, the second requirement is meaningful.

Input

The first line contains a single integer n ($2 \le n \le 10^5$), the number of vertices.

Next $n-1$ lines describe the trie. i-th of them contains two integers a and b, meaning that there is an edge between vertices a and b ($1 \le a, b \le n$).

Next $n-1$ lines describe the suffix links in the same format.

It is guaranteed that both the trie and the suffix links are trees and an answer exists.

Output

The first line should contain a single integer k ($1 \le k \le 3 \cdot 10^5$) — the number of the strings.

Next k lines should contain the strings. Each should start with a single integer l ($1 \le l \le 3 \cdot 10^5$), length of the string. l integers a_i ($1 \le a_i \le 3 \cdot 10^5$) representing the letters of the string should follow. Same a_i correspond to same letters and vice versa.

The total length of all strings should not exceed $3 \cdot 10^5$.

Examples

standard input	standard output
2 1 2 2 1	1 1 228
3 1 2 2 3 3 2 2 1	2 1 1 1 2
3 1 2 2 3 2 1 3 2	1 2 1 1
3 1 2 2 3 3 1 3 2	2 2 1 2 1 1
4 1 2 2 3 4 1 1 4 4 3 2 1	2 2 1 2 1 2
5 1 2 2 3 3 4 4 5 1 2 4 1 3 2 2 5	2 1 2 3 1 2 3
7 1 2 2 3 3 4 1 5 5 6 1 7 1 2 3 5 4 6 6 7 1 5 1 7	3 1 1 2 2 1 3 3 2 1

Problem B. Big Numbers

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds (4 seconds for Java)
Memory limit:	512 mebibytes

Problems with big numbers taken modulo some small number discriminate Python. © Some blue guy on Codeforces.

You are given a directed rooted tree. All edges are directed away from the root. Each edge has length which is a power of 2. The root of the tree has number 1.

Let's define two terms, which will depend on each other — a **trip** from vertex v, and a **journey** to vertex v.

A **journey** to some vertex v always starts from its parent p (it means that a journey to the root of the tree may never occur) and consists of three steps:

1. Traverse p-v edge (the edge from p to v).
2. Make a trip from v.
3. Teleport from vertex v to vertex p without traversing any edges.

A **trip** from vertex v is the following procedure:

1. Make a journey to all of the children of v (if any).
2. If v has at least one child, make a journey to some child of v one more time.

Note that any trip or journey always starts and ends at the same vertex.

Length of the trip is the total length of all traversed edges with multiplicity during the trip.

What is the maximum length of a trip from the root? Calculate it modulo 998 244 353.

Input

The first line of input contains a single integer n ($2 \le n \le 10^5$), the number of vertices in the tree.

Next $n-1$ lines describe the tree. i-th of these lines contains two integers p_i and c_i ($1 \le p_i \le i, 0 \le c_i \le 10^{18}$) describing an edge between vertices p_i and $i+1$ with length 2^{c_i}.

Output

Output a single integer, the maximum length of a trip from the root modulo 998 244 353.

Examples

standard input	standard output
7 1 1 2 1 1 4 2 2 1 2 5 0	52
3 1 28 1 1000000000000000000	752834992

Problem C. Cactus Revisited

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

«Contest should be comparable with regional competitions» - they say. Well, with this problem, this one should feel really NEERC.

A b-**fold coloring** of a graph G is an assignment of sets of colors of size b to vertices of G such that adjacent vertices are assigned with disjoint sets. An a:b-**coloring** is a b-fold coloring such that all assigned sets of colors are subsets of a universal set of size a.

A **cactus** is a connected graph in which every edge belongs to at most one simple cycle.

You are given a cactus. Find its a:b-coloring that minimizes the ratio $\frac{a}{b}$ among colorings with $1 \le b \le 1000$. If there are multiple suitable a:b-colorings with the smallest possible ratio, output any of them.

Input

The first line contains two integers n and m ($2 \le n \le 1000, 1 \le m \le 1500$), the number of vertices and the number of edges in the graph respectively.

Each of the next m lines contains two integers u and v ($1 \le u, v \le n$) describing an edge between vertices u and v.

It is guaranteed that the given graph is a cactus without loops and multiple edges.

Output

In the first line print two integers a and b ($1 \le a \le 10^6, 1 \le b \le 1000$). It can be proven, that in an optimal answer a will never exceed 10^6 under given limitations.

Each of the next n lines should contain b distinct numbers in the range from 1 to a. i-th of the lines should describe the set of colors assigned to the vertex i in an arbitrary order.

Examples

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

Problem D. Decent Sequence

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

The array a is **decent** if it can be split into two consecutive parts (possibly empty), such that the first one (the prefix) is non-decreasing, and the second (the suffix) is non-increasing. For example, $[1, 2, 3]$, $[1, 2, 2, 1]$ and $[3]$ are decent, while $[3, 2, 2, 8]$ is not.

There is an array a consisting of n integers, which is not given to you. You are, however, given n integers l_i and n integers r_i. It is known that for all indices i from 1 to n we have $l_i \le a_i \le r_i$. Is a decent?

You have to give one of the three answers:

1. a is definitely decent.
2. a is definitely not decent.
3. Neither of the above is the case.

Input

The first line contains a single integer n ($1 \le n \le 10^6$), the length of a.

Each of the following n lines contains two integers l_i and r_i ($1 \le l_i \le r_i \le 10^9$) describing the range containing the element a_i.

Output

Output "TAK" (without quotes) if a is definitely decent.

Output "NEIN" (without quotes) if a is definitely not decent.

Output an arbitary string consisting of at least five but not more than 100 lowercase or capital english letters otherwise. *If you output something really inappropriate you might get Presentation Error. Both of the possible answers in the samples are not considered inappropriate.*

Examples

standard input	standard output
3 3 4 2 5 1 1	TAK
4 3 4 1 3 1 2 3 10	NEIN
3 1 10 1 10 1 10	NEITHER
4 1 2 1 1 1 1 1 2	Secret

Problem E. Expected Cycle Size

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

TL;DR Permutation pattern is a permutation with 0 as a wildcard. You are given a permutation pattern. For each index, find its expected cycle size if a random permutation conforming to the pattern is chosen and output it modulo 998 244 353.

And now five times as long, three times as formal

A **permutation** is an array p of length n, such that $\forall_{i \neq j} : p_i \neq p_j, \forall_i : 1 \leq p_i \leq n$

The **product of permutations** p and q which have the same length, denoted as $p \cdot q$ is the permutation r of the same length as both p and q, such that $\forall_i : r_i = p_{q_i}$

The **power** p^k where p is a permutation and k is a positive integer is

1. p if $k = 1$
2. $p^{k-1} \cdot p$ otherwise

Cycle size of index i is the minimal positive integer k, such that $(p^k)_i = i$. It can be shown that such number always exists.

Permutation pattern is an array of length n such that $\forall_{i \neq j} : a_i = 0 \ or \ a_i \neq a_j, \forall_i : 0 \leq a_i \leq n$.

We say permutation p conforms to the pattern t if $\forall_i : p_i = t_i \ or \ t_i = 0$.

Let ans_i be the expected cycle size of index i in a random permutation conforming to the pattern given in input. You are to find ans_i modulo 998 244 353.

Taking a potentially non-integer number X modulo M is the following procedure:

Jury guarantees that X is equal to some irreducible fraction $\frac{P}{Q}$ where Q has an inverse modulo M. In that case X modulo $M = A$, where A is an integer between 0 and $M - 1$ inclusive and $P - QA$ is divisible by M. It can be shown that A is unique.

Input

The first line contains one integer n ($1 \leq n \leq 10^6$), the length of the permutation pattern.

The second line contains n space separated integers t_i ($0 \leq t_i \leq n$).

It is guaranteed that t is a permutation pattern.

Output

Output n integers. i-th of them must be equal to ans_i modulo 998 244 353.

Examples

standard input	standard output
5 2 3 4 5 0	5 5 5 5 5
2 0 0	499122178 499122178
6 3 0 1 6 5 2	2 3 2 3 1 3

Note

In the second example both ans_i are equal to $\frac{3}{2}$ which equals 499122178 modulo 998244353.

Problem F. Folding

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

Let's define an operation on a string that we will call **folding**. Folding consists of several (possibly zero) folds. Each fold happens between a pair of consecutive letters and places the second part of the string (the one further from the beginning of the string) above the first part (the one which is closer to the beginning of the string), running in the opposite direction and aligned to the position of the fold. Using this operation, a string is converted into a structure that has one more level than there were fold operations performed.

After the folding the string can be seen as several piles of letters. We call the folding **valid** if all letters in the same pile are the same.

The **corresponding set** of the folding is the set of positions of letters *in the original string* immediately before the positions of the folds. For example the corresponding set of the folding consisting of a single fold between 2nd and 3rd letter is $\{2\}$, and the corresponding set of the folding consisting of zero folds is an empty set. Two folding are to be considered the same if only if their corresponding set are the same.

A set S is **arithmetic** if integers a and b ($0 \le b < a$) can be chosen such that $x \in S \iff 1 \le x < n,\ x = b \pmod{a}$, where n is the length of the string. For example, if n is equal to 10, then the sets $\emptyset$, $\{2\}$, $\{1, 9\}$, $\{2, 4, 6, 8\}$ are considered arithmetic, while the sets $\{1, 2, 4\}$, $\{1, 5\}$, $\{7, 8, 9\}$ are not.

A folding is **awesome** if it is valid and its corresponding set is arithmetic. See pictures at the next page for clarity.

You are given a string. Count the number of awesome foldings of the given string.

Input

The only line contains a string consisting of Latin letters, digits, underscores or dash characters of length n ($1 \le n \le 10^6$).

Output

Output a single integer — the number of awesome foldings of the given string.

Examples

standard input	standard output
aaaaa	9
V-oo-V	2
gritukan	1
Lhic	1
000000000000000000000000	6
SpyCheessee	3
Aa	1
228322	4
--------------------------------------	380

Note

In the first example the corresponding sets of awesome foldings are: $\emptyset$, $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{1, 3\}$, $\{1, 4\}$, $\{2, 4\}$, $\{1, 2, 3, 4\}$.

```
  a
cba
cba
  a
```

This is an awesome folding of the string "aabccbaa". The corresponding set is $\{1, 4, 7\}$.

```
abc
```

This is an awesome folding of the string "abc". The corresponding set is $\emptyset$.

```
   k
ghij
fedc
  ab
```

This is not a valid folding, but it is a folding and it has an arithmetic corresponding set. The corresponding set is $\{2, 6, 10\}$.

```
 cdefgh
ab
```

This is not a folding, because the upper part runs in the same direction as the lower part.

```
  c
ab
```

This is not a folding, because the upper part is either running in the same direction (if we consider it running to the right) or it is not aligned (if we consider it running to the left).

```
 aaaa
 aa
aaaa
```

This is a valid folding, but it's corresponding set is not arithmetic. The corresponding set is $\{4, 6\}$.

And that is how you tex.

Problem G. Game

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

Those guys are definitely teaming.

n players numbered from 0 to $n-1$ are playing a game. There is a number x, which is initially equal to 0. There are n numbers a_i ($0 \le i, a_i \le n-1$) that are subject to change between the rounds of the game. The game proceeds as follows:

1. Player 0 either skips his turn or makes x equal to $(x + a_0) \bmod n$.
2. Player 1 either skips his turn or makes x equal to $(x + a_1) \bmod n$.
3. ...
4. Player $n-1$ either skips his turn or makes x equal to $(x + a_{n-1}) \bmod n$.

After this process, the player with the number x wins.

Each player makes a move (that is, changes x) if and only if he will win if he makes a move, but won't win if he doesn't. Players know that everyone plays according to this strategy.

You have to answer q queries: if we change a_x to y who will win the game? Note that the changes are **not** reverted after each query.

Input

The first line of input contains a single integer n ($1 \le n \le 10^5$) — the number of players.

The second line contains n integers — initial values of a_i ($0 \le a_i \le n-1$).

The third line contains a single integer q ($0 \le q \le 10^5$) — the number of queries.

q lines follow. i-th of them contains two integers x_i and y_i ($0 \le x_i, y_i \le n-1$) meaning that a_{x_i} is equal to y_i from this query onwards.

Output

Output $q+1$ integers. i-th them should be the number of the winner of the game after $i-1$ queries.

Examples

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

Problem H. Hidden Pool

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

Let's choose a time moment as the beginning of time, which we will refer to as simply The Beginning.

The matchmaking of a well-known game Rubbish Bin 7 during a time interval of 10^{100} seconds starting from the beginning (the time interval will be referred to as The Interval) has the following model:

Every match contains exactly p players. Let's assume that the number of players searching for a match during The Interval is divisible by p. Sort the players by the time they started searching for a match, with ties broken arbitrarily. The i-th match contains players from $(i-1)p+1$-th to ip-th in this order.

Arthas is about to stream himself playing Rubbish Bin 7 and is going to start searching for a match kT seconds after the beginning where T is a fixed given integer, and k is a non-negative integer Arthas is free to choose.

There are also n other normal players. i-th of them will start searching for a match t_i seconds after The Beginning. t_i is not divisible by T.

There are also m streamsnipers whose goal is to get in the same match as Arthas *(and turn some quality content into Arthas being tilted and whining all game long)*. All of them must search for a match exactly once during The Interval and they have to do it $l+0.5$ seconds after the beginning (l is some integer a streamsniper can choose for himself, but different streamsniper can start searching at the same moment). Streamsnipers know when the normal players will start searching (i.e. the values of t_i). Arthas's stream has a delay of d seconds, so d seconds after he starts searching for a match the streamsnipers will know it. Arthas is going to start searching for a match during The Interval *(You may assume he's not going to get DDoSed)*.

Note that under the limitations described above the way ties between players that started searching at the same time are broken doesn't matter (since normal players are indistinguishable, as well as streamsnipers).

Help streamsnipers design a strategy than will maximize the minimum amount of streamsnipers in a match with Arthas among all possible times when Arthas might start searching for a match. Find that amount of streamsnipers.

Input

The first line of input contains 5 integers n, m, T, d, p ($1 \le n \le 5 \cdot 10^5$, $1 \le m, T, d, p \le 10^{12}$, $n+m+1$ is divisible by p) — the number of normal players, the number of streamsnipers, the multiple for Arthas's searching time, the delay on the stream, and the number of players in a single match.

The second line contains n integers t_i ($1 \le t_i \le 10^{18}$, t_i is not divisible by T). i-th of them is the time when i-th normal player starts searching.

Output

Output a single integer — the amount of streamsnipers you've been asked to find.

Examples

standard input	standard output
8 3 6 3 3 13 5 21 23 25 22 7 11	1
6 5 7 7 4 1 2 3 4 5 6	2

Note

In the first example one possible strategy is as follows: one streamsniper starts searching at 5.5, another at 11.5, and the last one starts searching 3.5 seconds after Arthas does (he will know because the delay is 3, which is less than 3.5).

Problem I. Intersection Of Tangents

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

In this problem polygons are assumed to have no self-touchings or self-intersections.

A **tangent** to a polygon is a straight **line** that contains at least one point on the boundary of the polygon, and none of its interior points.

You are given a polygon with integer vertex coordinates. The polygon is not necessarily convex. Find a point with integer coordinates such that there exist two tangents to this polygon which both pass through this point and intersect at 90°. It is guaranteed that at least one solution exists. If there are multiple solutions, output any of them.

Input

The first line of input contains a single integer n ($3 \le n \le 1000$) — the number of vertices in the polygon.

n lines follow describing the vertices of the polygon. i-th of them contains two integers x_i and y_i ($-10^8 \le x_i, y_i \le 10^8$) — the coordinates of i-th vertex. The vertices are given in counter-clockwise order.

The polygon has no self-touchings or self-intersections. There are no three consecutive points which lie on the same line.

It is guaranteed that an answer exists.

Output

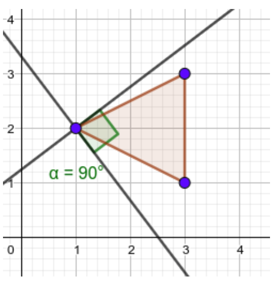
Output two integers x and y ($-10^9 \le x, y \le 10^9$) — the coordinates of the point you found.

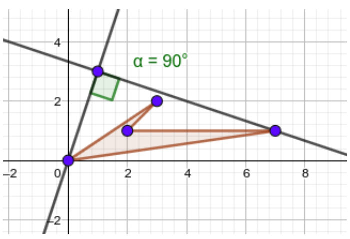
Examples

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

Note

Images for samples and some random photo.





Problem J. Just Another Edge

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

A **simple undirected graph** is an undirected graph that doesn't contain loops or multiple edges.

A **planar graph** is a simple undirected graph such that it can be embedded in the plane, i.e., it can be drawn on the plane in such a way that its edges intersect only at their common endpoints. In other words, it can be drawn in such a way that no edges cross each other.

An **independent set** is a set of vertices in a graph, no two of which are adjacent.

A **tripartite graph** is a simple undirected graph whose vertices can be divided into three disjoint independent sets.

You are given a planar graph. You are about to add a single edge to it. You know there is no way to add an edge, such that the resulting graph is planar. How many ways are there to add an edge such that the resulting graph is tripartite?

Note that you can not add multiple edges or loops because the resulting graph must be simple. Edges $a - b$ and $b - a$ are the same and should be counted once.

Input

The first line of input contains two integers n and m ($3 \le n, m \le 3 \cdot 10^5$) — the number of vertices and the number of edges respectively.

m lines follow. Each of them contains two integers a and b ($1 \le a, b \le n, a \ne b$), meaning that there is an edge between vertices a and b.

It's guaranteed that the graph conforms to the conditions described above.

Output

Output a single integer — the number of ways to add an edge, such that the resulting graph is tripartite.

Examples

standard input	standard output
3 3 1 2 1 3 2 3	0
4 6 1 2 3 1 4 1 2 3 3 4 2 4	0
8 18 1 2 1 3 1 4 1 5 1 6 1 7 2 3 3 4 4 5 5 6 6 7 2 7 8 2 8 3 8 4 8 5 8 6 8 7	3

Note

Isn't the last sample sample beautiful and concise?

Problem K. Khalin Graph

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

Little George Khalin designed a new type of graphs. He doesn't like the letter K, so he dropped it.

Preorder of a tree (sometimes called **time-in order**) is obtained using the following procedure:

Let's fix the root of the tree, and direct all edges away from the root. Preorder of the subtree of a vertex v is v followed by preorders of subtrees of all its children (if any) in some order. A preorder of a tree with a fixed root is any preorder of the subtree of the root.

Note that there are multiple preorders of the same tree, since a preorder depends on the choice of the root, as well as the order in which children subtrees are considered at every vertex.

A **Halin graph** is a graph obtained using the following procedure:
There is a tree that we will call the **base tree** of the graph, which has at least 4 vertices and has no vertices of degree 2. One of its preorders is specified. The root of the tree with respect to this preorder is not a leaf.

Let $v_1, v_2, ..., v_m$ be the leaves of the tree in order they appear in the preorder. For each i from 1 to m, add an edge between the vertices v_i and $v_{(i \bmod m)+1}$ to the tree. Those edges are called **additional**. The resulting graph is the Halin graph with respect to the base tree and the specified preorder.

A **3-matching** of a graph G is a set of edges S such that the connected components of the graph formed by removal of all edges not in S from G are trees of size 3 or 1.

You are given a Halin graph. Find the number of its 3-matchings modulo 998 244 353.

Input

The first line contains a single integer n ($4 \le n \le 10^5$) the number of vertices in the base tree. Vertices are enumerated according to the preorder.

The second line contains $n-1$ integers. i-th of them is p_i ($1 \le p_i \le i$), describing an edge between p_i-th and $i+1$-th vertices in the base tree.

It is guaranteed that the base tree is a tree, has no vertices of degree 2, and that the vertex 1 is not a leaf.

Output

Output a single integer — the number of 3-matchings of the given graph modulo 998 244 353.

Examples

standard input	standard output
4 1 1 1	13
6 1 1 3 3 1	34
11 1 1 3 4 4 3 3 1 9 9	737

Note

In the first example the actual Halin graph is the complete graph on four vertices.

In the second example the leaves are $[2, 4, 5, 6]$, thus there are four additional edges — (2, 4), (4, 5), (5, 6), (6, 2).

Note that there is no letter K in the meaningful parts of the statement.

Problem L. Lysergic Acid Diethylamide

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

might or might not had been used during making of this problem.

In this problem **functions** are implictly assumed to have a single non-negative integer as an argument and produce a single non-negative integer as a result.

f is a function. $f(x) = 1 + 2 + \ldots + x$. More formally, $f(x)$ is the sum of all positive integers less than or equal to x.

s_k is a family of functions. s_0 is an identity function ($s_0(x) = x$) and $s_k(x) = s_{k-1}(f(x) + k)$.

You are given t test cases. i-th test case contains three integers x_i, k_i and p_i. For each test case find an integer m_i such that $-1 \le m \le p_i - 1$, $s_{k_i}(x_i) \bmod p_i \not\equiv m_i$. You may use $m_i = -1$ no more than 20 times. p_i are pairwise distinct. Note, that $a \bmod p \ge 0$, where a is an arbitrary integer, so $m_i = -1$ is correct for any particular test case.

Why would you do that? It's simple. Finding correct answers is easy and conformist. On the other hand, finding incorrect answers is challenging and original. However, in this problem it's a bit too challenging, because of $p = 1$ case. So you decided, that you will skip some test cases with $m_i = -1$ wildcard. Sounds reasonable (not).

Input

The first line of input contains a single integer t ($1 \le t \le 5000$) — the number of test cases.

t lines follow. i-th of them contains three integers x_i, k_i, p_i ($1 \le x_i \le 10^9, 0 \le k_i \le 10^5, 1 \le p_i \le 10^4$).

It is guaranteed that $\forall_{i \ne j}\ p_i \ne p_j$.

Output

Output t integers. i-th of them should be equal to m_i.

The number of indices i such that $m_i = -1$ should not exceed 20.

Examples

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

Problem M. Moving Randomly

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

Consider a following game on an array:

You are playing as a pointer. Initially you are pointing to a random element of this array (equiprobably).

At each moment of the game you may do one of the following:

1. **Finish the game**. The game ends and your score is equal to the value of the element you are pointing to.
2. **Move**. You equiprobably move one element to the left or to the right. If you may point outside of bounds of the array after this you are not allowed to choose this option. (*You might get dereferenced and then you might turn into a goat, the whole game might be optimized out, e.t.c — it's undefined behaviour, who knows. Anyway you want neither of these.*)

You make this choice repeatedly until the game ends. It can be proven that $\lim_{m\to\infty} f(m) = 0$, where $f(m)$ is the probability that you can choose Move m times.

The score of the array is the maximum expected score you can obtain if you play optimally. (*You are a smart pointer.*)

You are given an array a. For each of its prefixes calculate it's score modulo 998 244 353.

Taking a potentially non-integer number X modulo M is the following procedure:

Jury guarantees that X is equal to some irreducible fraction $\frac{P}{Q}$ where Q has an inverse modulo M. In that case X modulo $M = A$, where A is an integer between 0 and $M-1$ inclusive and $P-QA$ is divisible by M. It can be shown, that A is unique.

Input

The first line contains one integer n $(1 \le n \le 5 \cdot 10^5)$ — length of a.

The second line contains n space separated integers a_i $(1 \le a_i \le 10^6)$ — elements of a.

Output

Output n integers. i-th of them should be equal to the score of prefix of a of length i taken modulo 998244353.

Examples

standard input	standard output
3 3 1 2	3 2 499122179
6 6 1 2 5 3 4	6 499122180 4 499122182 5 582309211

Note

Consider the prefix of length 3 of the first example (i.e the full array), which is equal to [3,1,2]. The optimal strategy is to move from second element if you start pointing at it. You won't to be able to move afterwards or if you start poining to some element other than second.

The score is $\frac{5}{2}$ which is equal to 499122179 modulo 998244353.