Problem A. Planting Trees

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Little L likes planting trees.

Little L planted d trees on both the left and right sides of the yard. On the left side, the i-th tree has a height of x_i meters on the first day and grows f_i meters each day. On the right side, the i-th tree has a height of y_i on the first day and grows g_i meters each day.

However, Little L does not want the trees to be too high. To keep things neat, for the tree numbered i, if its height reaches or exceeds m_i, he will prune it by m_i meters until its height is less than m_i.

Now, Little L wants to know, for the tree numbered i, during the first n_i days after planting, after performing pruning each day, how many days the left tree is shorter than the right tree.

Note: pruning is also performed on the first day.

Input

There is only one test case in each test file.

The first line contains a positive integer d ($1 \le d \le 10^4$), denoting the number of trees on each side of the yard.

Each of the next d lines contains six positive integers $f_i, x_i, g_i, y_i, n_i, m_i$ ($1 \le f_i, x_i, g_i, y_i, n_i, m_i \le 10^6$), with the meanings described above.

Output

Output d lines, each line containing a non-negative integer, representing the number of days within n_i days that the left tree is shorter than the right tree for the tree numbered i.

Example

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

Note

For the first sample, we count the first 3 days:

Day 1: The initial height of the tree on the left side of the road is 4 meters, and the tree on the right is 3 meters. The left tree exceeds the 4-meter limit, so it is pruned. After pruning, the left tree is 0 meters and the right tree is 3 meters; the left is lower than the right.

Day 2: The left tree is 7 meters and the right tree is 9 meters. They both exceed 4 meters, so they are pruned. After pruning, the left tree is 3 meters and the right tree is 1 meter; the left is not lower than the right.

Day 3: The left tree is 10 meters and the right tree is 7 meters. They both exceed 4 meters, so they are pruned. After pruning, the left tree is 2 meters and the right tree is 3 meters; the left is lower than the right.

Therefore, the output is 2.

Problem B. 77G Network

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

The map of Meowland can be regarded as a rooted tree with n nodes and $n-1$ edges, where nodes represent cities and an edge represents a road connecting two cities. The capital is at the root, node 1.

Now Meowland wants to install brand-new 77G network base stations in some cities. To ensure that every road has sufficient network quality, at least one of the endpoint cities of each road must have a base station. However, due to unknown technical issues, the network system has m faults. Fault i can be described as follows: if a base station is installed at city x_i, then all cities within the subtree of x_i whose distance to x_i is exactly y_i cannot install base stations; otherwise, the stations will interfere with each other and become unusable.

Please help Meowland choose some cities so that installing base stations in those cities satisfies all the above requirements. If no valid scheme exists, output "No".

Input

Each test file contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 10^3$). The description of the test cases follows.

The first line of each test case contains an integer n ($2 \le n \le 2 \times 10^5$), denoting the number of cities.

The second line contains $n-1$ integers $f_2, f_3, \ldots, f_n$ ($1 \le f_i < i$), where f_i denotes the parent of city i.

The third line contains an integer m ($0 \le m \le 2 \times 10^5$), denoting the number of faults.

Each of the next m lines contains two integers x_i and y_i ($1 \le x_i, y_i \le n$), denoting the i-th fault. There may exist two identical faults.

For each test file, it is guaranteed that the sum of n over all test cases does not exceed 2×10^5, and the sum of m over all test cases does not exceed 2×10^5.

Output

For each test case, if there is no valid scheme, output a line "No".

Otherwise, output three lines representing your scheme:

- On the first line output "Yes".
- On the second line output an integer c, denoting the number of cities you choose.
- On the third line output c distinct integers, denoting the cities you choose.

If there are multiple valid schemes, you may output any one. When outputting the scheme, you may output the c cities in any order.

Example

standard input	standard output
3 6 1 1 2 2 4 2 1 3 2 1 6 1 1 2 2 4 6 1 1 1 2 1 3 2 1 2 2 2 1 8 1 1 3 1 1 2 4 10 1 2 3 2 1 3 5 1 6 1 3 2 4 1 3 2 5 1 3 1	Yes 4 5 3 4 1 No No

Problem C. Not Aqre

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Given an $n \times m$ matrix, where $n \times m$ is divisible by 3. You need to fill the matrix with 0, 1, and 2, each appearing exactly $\frac{nm}{3}$ times, such that:

- No three identical numbers appear consecutively in a horizontal or vertical direction.
- All cells filled with 0 form an 8-connected region[†], all cells filled with 1 form an 8-connected region, and all cells filled with 2 form an 8-connected region.

[†] Two cells are 8-adjacent if they share a side or a corner. A set of cells is 8-connected if for any two cells in the set, there exists a path connecting them entirely within the set, and each step moves to an 8-adjacent cell.

Input

Each test file contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 10^3$). The description of the test cases follows.

The first line of each test case contains two integers n and m ($1 \le n, m \le 10^3$, $n \times m$ is divisible by 3), representing the number of rows and columns of the matrix.

For each test file, it is guaranteed that the sum of $n \times m$ over all test cases does not exceed 10^6.

Output

For each test case, if it is impossible to construct a matrix satisfying the conditions, output a single line "No".

Otherwise, output "Yes" on the first line, followed by the matrix with n lines, each containing m numbers. If there are multiple solutions, you may output any one of them.

Example

standard input	standard output
2 3 4 1 12	Yes 0012 0021 2211 No

Problem D. Outsmarting

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Given an undirected bipartite graph $G = (U, V, E)$, where U and V are sets of vertices of the two parts and $E \subseteq U \times V$ is the set of edges. Some vertices are marked as "disabled", while the others are "enabled". There is a piece on the graph. Two players, the first player and the second player, move the piece alternately. The first player starts from vertex 1 in the left part U of the bipartite graph. The movement rules are as follows:

- On a player's turn, if the piece is on vertex x, the player must choose an edge (x, y) such that vertex y is enabled;
- The player moves the piece to y and marks x as disabled;
- If a player cannot find a movable edge, that player loses and the opponent wins.

Before the game starts, a "flip" operation can be applied to some vertices: changing an enabled vertex to disabled, or a disabled vertex to enabled. Each vertex can be flipped at most once, and the starting vertex cannot be disabled.

Determine the minimum number of flips needed to guarantee a win for the first player, assuming both players play optimally. If it is impossible for the first player to win, output "-1".

Input

There is only one test case in each test file.

The first line contains three integers n_U, n_V and m ($1 \le n_U, n_V \le 100$, $0 \le m \le n_U \cdot n_V$), representing the size of the left part U, the size of the right part V, and the number of edges connecting the two parts, respectively.

The next m lines each contain two integers u and v ($1 \le u \le n_U$, $1 \le v \le n_V$), representing an edge connecting vertex u in the left part and vertex v in the right part.

The next line contains n_U integers $s_{u,1}, s_{u,2}, \ldots, s_{u,n_U}$ ($s_{u,i} \in \{0, 1\}$, $s_{u,1} = 1$). If $s_{u,i} = 0$, the i-th vertex in the left part is disabled; otherwise, it is enabled.

The next line contains n_V integers $s_{v,1}, s_{v,2}, \ldots, s_{v,n_V}$ ($s_{v,i} \in \{0, 1\}$). If $s_{v,i} = 0$, the i-th vertex in the right part is disabled; otherwise, it is enabled.

Output

Output a single integer representing the minimum number of flips required to guarantee a win for the first player. If it is impossible to guarantee a win, output "-1".

Examples

standard input	standard output
4 4 8 1 1 2 1 2 2 3 1 3 3 4 2 4 3 4 4 1 1 1 1 1 1 1 0	1
4 2 5 1 1 2 1 2 2 3 2 4 2 1 1 1 1 1 1	1
1 1 0 1 0	-1

Problem E. Magic Trick

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

The magician Link has a deck s consisting of n cards, each with an integer on it representing its value.

Link can pick two cards from the deck. Let the values of the cards be x and y. If $|x - y| = 1$, he can perform one of the following two tricks:

- Change the two card values to $x - 1$ and $y - 1$, then put them back into the deck;
- Change the two card values to $x + 1$ and $y + 1$, then put them back into the deck.

You also have a deck t containing n cards. You want to know whether Link can make his deck identical to yours after performing the magic trick any number of times (possibly 0). Please note that there can be cards with the same value in the deck, and there is no difference between those cards.

Input

Each test file contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 10^4$). The description of the test cases follows.

The first line of each test case contains an integer n ($1 \le n \le 10^5$), denoting the number of cards in the deck.

The second line contains n integers $s_1, s_2, \cdots, s_n$ ($1 \le s_i \le 10^9$), representing the values of Link's cards.

The third line contains n integers $t_1, t_2, \cdots, t_n$ ($1 \le t_i \le 10^9$), representing the values of your cards.

For each test file, it is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, output "`Yes`" if Link can make his deck identical to yours after performing the magic trick any number of times (possibly 0); otherwise, output "`No`".

Example

standard input	standard output
3 3 1 10 100 100 10 1 2 1 2 99 100 3 1 2 3 1 1 6	Yes Yes No

Note

In the first sample test case, the two decks are already identical, so the output is "`Yes`".

In the second sample test case, by performing the second magic trick 98 times consecutively on two cards, Link can obtain the target deck, so the output is "`Yes`".

In the third sample test case, it is impossible to transform Link's deck into the target deck, so the output is "`No`".

Problem F. Divide and Conquer

Input file:	standard input
Output file:	standard output
Time limit:	2 seconds
Memory limit:	1024 megabytes

Fall has an array of length n and q queries on this array. Each query is an interval $[l_i, r_i]$.

Fall decides to handle these queries using a divide-and-conquer approach. Initially, all queries to be processed lie within the interval $[1, n]$, which is at layer 1. Suppose the current set of queries to be processed lies within $[L, R]$ at layer d. He may choose a position x from $[L, R]$ and process all queries whose intervals contain x at layer d. The remaining queries lie in $[L, x-1]$ and $[x+1, R]$, which can be handled recursively as two subproblems at layer $d+1$ (if there are still queries in those intervals).

You are asked to minimize the maximum layer number at which any query is processed.

Input

Each test file contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n and q ($1 \le n, q \le 10^6$), denoting the length of the array and the number of queries.

Each of the next q lines contains two integers l_i and r_i ($1 \le l_i \le r_i \le n$), representing a query.

For each test file, it is guaranteed that the sum of n over all test cases does not exceed 10^6, and the sum of q over all test cases does not exceed 10^6.

Output

For each test case, output one line containing a single integer: the minimum possible value of the maximum layer number at which the queries are processed.

Example

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

Note

For the first sample test case, one optimal divide-and-conquer strategy is: choose $x = 3$, process the first query at layer 1, recurse the second and third queries into the interval $[1, 2]$, and recurse the fourth and fifth queries into the interval $[4, 5]$. Both child intervals require one more recursion, so the maximum layer at which queries are processed is 3.

Problem G. Projection

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Given an n-dimensional hypersphere and hyperrectangle, defined as follows:

Hypersphere: $B \triangleq \{\mathbf{x} \mid \sum_{i=1}^{n} x_i^2 \le R^2; \mathbf{x} \in \mathbb{R}^n\}$;

Hyperrectangle: $C \triangleq \{\mathbf{x} \mid |x_i| \le l_i, \forall i \in [1, n]; \mathbf{x} \in \mathbb{R}^n\}$.

Let $S \triangleq B \cap C$. Given a point $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, find the point in S (including the boundary) that is closest to $\mathbf{x}$, denoted by $\mathbf{y} = (y_1, y_2, \ldots, y_n)$.

A point $\mathbf{p}$ is considered closer to $\mathbf{x}$ than a point $\mathbf{q}$ if and only if $\sum_{i=1}^{n}(x_i - p_i)^2 \le \sum_{i=1}^{n}(x_i - q_i)^2$.

Input

Each test file contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 3 \times 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n and a ($2 \le n \le 10^5$, $1 \le a \le 10^9$), where a is the value of R^2.

The second line contains n integers $x_1, x_2, \ldots, x_n$ ($|x_i| \le 10^9$).

The third line contains n integers $l_1, l_2, \ldots, l_n$ ($1 \le l_i \le 100$).

For each test file, it is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, output one line containing n real numbers $y_1, y_2, \ldots, y_n$.

Your answer is considered correct if its absolute or relative error does not exceed 10^{-6}.

Formally, let your answer be y_i, and the jury's answer be y_i^*. Your answer is accepted if and only if $\frac{|y_i - y_i^*|}{\max(1, |y_i^*|)} \le 10^{-6}$ for all $i \in [1, n]$.

Example

standard input	standard output
3 2 25 1 10 4 4 2 25 10 1 4 4 3 6 3 3 3 2 2 2	1.0000000000 4.0000000000 4.0000000000 1.0000000000 1.4142135624 1.4142135624 1.4142135624

Problem H. Restore the Array

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

This is an interactive problem.

Little L has a hidden set of size n, $\{a_1, a_2, \ldots, a_n\}$ ($0 \le a_i < 2^{30}$), with all elements distinct.

To guess the values of the elements in the set, you can make queries to Little L. In a query, you give a non-negative integer c, and Little L will tell you $\max\{a_i \oplus c\}$, where $\oplus$ denotes bitwise XOR [†].

For example, if $a = \{1, 4, 6\}$ and $c = 3$, Little L will respond 7, because $\max\{1 \oplus 3, 4 \oplus 3, 6 \oplus 3\} = \max\{2, 7, 5\} = 7$.

You need to guess all elements of the set using no more than $2n - 2$ queries.

[†] Bitwise XOR is defined as adding two binary numbers bit by bit modulo 2. For example: $(0011)_2 \oplus (0101)_2 = (0110)_2$.

Input

Each test file contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 10^4$). The description of the test cases follows.

Each test case contains a single line with a positive integer n ($2 \le n \le 10^5$), representing the size of the set.

For each test file, it is guaranteed that the sum of n over all test cases does not exceed 10^5.

Interaction Protocol

To make a query, output in the format "`?` c" ($0 \le c < 2^{30}$). After flushing the output, your program should read a single integer from input, which is the value of $\max\{a_i \oplus c\}$. For each test case, you can make at most $2n - 2$ queries. If you exceed this number of queries, the verdict will be **undefined**.

To guess the set, output in the format "`!` a_1 a_2 $\ldots$ a_n" ($0 \le a_i < 2^{30}$). The order of elements can be arbitrary. After flushing the output, the interaction for the current test case ends. You should immediately start handling the next test case, or exit if all test cases have been processed.

Note that the interactor is **adaptive**. This means that the elements in the set are not fixed at the beginning but may be dynamically adjusted based on your queries. The interactor guarantees that adjustments will not invalidate previous responses.

To flush your output, you can use:

- `fflush(stdout)` (if you use `printf`) or `cout.flush()` (if you use `cout`) in C and C++.
- `System.out.flush()` in Java and Kotlin.
- `stdout.flush()` in Python.

Example

standard input	standard output
1 4 7 7 7 7	 ? 0 ? 1 ? 5 ? 6 ! 1 2 6 7

Problem I. Dolls 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Alice has just obtained n Russian dolls of distinct sizes. She arranges these dolls in a row from left to right, with the i-th position containing a doll of size a_i.

Let l_i and r_i denote the size of the smallest and largest doll in the i-th cell, respectively. Two adjacent cells i and $i+1$ can be merged if and only if $r_i < l_{i+1}$ or $r_{i+1} < l_i$. The new doll will contain all dolls from the original cells i and $i+1$, and will be placed in cell i. All dolls in cells with indices greater than $i+1$ will shift one position to the left to fill the gap.

For example, when $n = 5, a = [1, 2, 3, 5, 4]$, Alice can:

1. Merge the dolls in cells 1 and 2. The remaining dolls in the four cells have sizes $[(1, 2), (3), (5), (4)]$.
2. Merge the dolls in cells 2 and 3. The remaining dolls in the three cells have sizes $[(1, 2), (3, 5), (4)]$.
3. Merge the dolls in cells 1 and 2. The remaining dolls in the two cells have sizes $[(1, 2, 3, 5), (4)]$. According to the rules, the dolls in the remaining two cells can no longer be merged.

Alice will continue performing merge operations until no further merges are possible. What is the minimum number of merge operations Alice has to perform?

Input

Each test file contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 10^4$). The description of the test cases follows.

The first line of each test case contains an integer n ($2 \le n \le 10^6$), denoting the number of dolls.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$, $\forall i \ne j, a_i \ne a_j$), representing the sizes of the dolls at each position initially.

For each test file, it is guaranteed that the sum of n over all test cases does not exceed 10^6.

Output

For each test case, output a single integer on one line, representing the minimum number of merge operations required.

Example

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

Problem J. Circular Matching

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Given two strings s and t, both consisting of lowercase letters.

Let $s_{l,r}$ be the substring of s from its l-th character to its r-th character with length $L = r - l + 1$. We say that t appears on the circle of $s_{l,r}$ if and only if there exists a starting position p ($0 \le p < L$) such that for all $0 \le k < |t|$, we have $s_{l,r}[(p + k) \bmod L] = t[k]$. If $L < |t|$, then t is considered not to appear on the circle of $s_{l,r}$.

Count how many distinct pairs (l, r) satisfy that t appears on the circle of $s_{l,r}$.

Input

Each test file contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 2 \times 10^5$). The description of the test cases follows.

The first line of each test case contains a string s ($1 \le |s| \le 2 \times 10^5$).

The second line contains a string t ($1 \le |t| \le 2 \times 10^5$).

It is guaranteed that all strings consist of lowercase letters.

For each test file, it is guaranteed that the sum of $|s|$ over all test cases does not exceed 2×10^5, and the sum of $|t|$ over all test cases does not exceed 2×10^5.

Output

For each test case, output one integer on a single line, representing the number of pairs (l, r) that satisfy the condition.

Examples

standard input	standard output
3 cabcccabc ccabcc bcdabc abcd aaaxaaaxaaa axaaaxaa	3 3 10
4 aaaaaaaaa a abacab caba baaabaab aabaaab bbabbbabb bbbabbb	45 4 2 3

Problem K. Organize the Bookshelf

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Given two sequences a and b of length n, you may perform the following operation on the two sequences at most $3n$ times:

Choose any integers i and j ($1 \le i, j \le n$), then swap a_i and b_j. The cost of this operation is $|i - j|$.

Find a sequence of operations that makes a and b equal while minimizing the total cost. If it is impossible to make the two sequences equal by the operations above, output "-1".

Input

Each test file contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 2 \times 10^5$). The description of the test cases follows.

The first line of each test case contains an integer n ($1 \le n \le 2 \times 10^5$), denoting the length of the sequences.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$), representing the sequence a.

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le n$), representing the sequence b.

For each test file, it is guaranteed that the sum of n over all test cases does not exceed 2×10^5.

Output

For each test case, output on the first line two integers *cost* and *num* ($0 \le num \le 3n$), representing the minimum total cost and the corresponding number of operations. If it is impossible to make the two sequences equal by the operations above, output "-1".

Then output *num* lines, each containing two integers x_i and y_i ($1 \le x_i, y_i \le n$), indicating that in the i-th operation you swap a_{x_i} and b_{y_i}.

It can be proven that if it is possible to make the two sequences equal by the above operations, then there exists an optimal solution with at most $3n$ operations.

Example

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

Problem L. ICPC

Input file:	standard input
Output file:	standard output
Time limit:	6 seconds
Memory limit:	1024 megabytes

Fall is taking part in the International Copy & Paste Challenge (ICPC). Suddenly, another contestant behind him stood up and tore his answer sheet into two pieces. Fall is very sad because he would have to spend time and ink to rewrite his answers. Unless... all his answers happen to lie entirely on one of the two halves?

Without time to grieve, Fall immediately begins to compute the probability that all answers lie entirely on one half of the sheet. Formally, the answer sheet can be divided into an $n \times m$ grid of unit squares. There are k segments of Fall's handwriting on the sheet. The i-th handwriting segment can be represented as the rectangle enclosed by (a_i, b_i), (a_i, d_i), (c_i, d_i), (c_i, b_i), where $0 \le a_i < c_i \le n, 0 \le b_i < d_i \le m$.

The contestant tore Fall's answer sheet along a "monotone path", and this path is chosen uniformly at random from all monotone paths. A monotone path is a lattice path from $(0, 0)$ to (n, m) where each step moves either right (by $+1$ in the x direction) or up (by $+1$ in the y direction). For convenience, even if such a path does not split the sheet into two halves (i.e., one half is empty), that path is still considered one of the monotone paths.

Having said that, Fall has computed the probability that all handwriting lies on one half. He wants to verify it, so please compute this probability as well. To prevent that contestant from tearing the sheet again after seeing the probability, report the answer modulo 998 244 353 †.

† Formally, let $M = 998\,244\,353$. It can be shown that the answer can be represented as a reduced fraction $\frac{p}{q}$ where p and q are integers and $q \not\equiv 0 \pmod{M}$. Output the integer equal to $p \cdot q^{-1} \bmod M$. In other words, output an integer x such that $0 \le x < M$ and $x \cdot q \equiv p \pmod{M}$.

Input

There is only one test case in each test file.

The first line contains three integers n, m, k ($1 \le n, m, k \le 2 \times 10^5$), representing the size of the paper and the number of key regions.

Each of the next k lines contains four integers a_i, b_i, c_i, d_i ($0 \le a_i < c_i \le n$, $0 \le b_i < d_i \le m$), describing a key region.

Output

Output a single integer representing the probability that all handwriting lies on one half of the sheet modulo 998 244 353.

Examples

standard input	standard output
4 3 3 0 0 1 1 1 1 3 2 3 2 4 3	199648871
200000 200000 2 114 514 1919 810 233 666 77777 88888	243245654

Note

In the first sample test case, the answer in fraction form is $\frac{14}{35}$.

Problem M. Kingdom of Construction

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Given a sequence of length n, $a_1, a_2, \ldots, a_n$, you need to construct an $n \times n$ matrix A such that for any two sets $S \subseteq S' \subseteq [n]$,

$$\det(A_{S'}) - \det(A_S) \equiv \left(\sum_{i \in S'} a_i - \sum_{i \in S} a_i \right) \pmod{998\,244\,353}.$$

Here $[n]$ denotes the set $\{1, 2, \ldots, n\}$, and $\det(A)$ denotes the determinant of A. For a subset $T \subseteq [n]$, A_T denotes the principal submatrix of A formed by rows and columns with indices in T. By convention, $\det(A_\emptyset) = 1$.

Input

There is only one test case in each test file.

The first line contains an integer n ($1 \le n \le 18$), denoting the length of the sequence.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i < 998244353$), representing the elements of the sequence.

Output

Output n lines, with the i-th line containing n integers $A_{i,1}, A_{i,2}, \ldots, A_{i,n}$ ($0 \le A_{i,j} < 998244353$), representing the matrix A you constructed.

Example

standard input	standard output
2 6 8	7 12 4 9