Problem A. AND

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

You had an array a. After that, you calculated bitwise ANDs of all subarrays of the original array. Formally, you calculated all numbers of the form a_i AND a_{i+1} AND ... AND a_j for $1 \le i \le j \le \text{length}(a)$.

You remember the resulting set of all these numbers: a number lies in this set if and only if it can be represented as bitwise AND of at least one subarray. Sadly, you forgot the original array.

Find any array a which would produce the given set of ANDs on subarrays, or determine that there is no such array.

Input

The first line contains a single integer t ($1 \le t \le 10^5$), the number of test cases.

The first line of each test case contains a single integer n ($1 \le n \le 10^5$), the size of the given set.

The second line of each test case contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i \le 2^{20} - 1$), the elements of the set. It is guaranteed that all elements are distinct.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, if there is no such array, output -1.

Otherwise, on the first line, output the size of the original array k ($1 \le k \le 5n$).

On the next line, output k integers $a_1, a_2, \ldots, a_k$ ($0 \le a_i \le 2^{20} - 1$), the elements of the array.

If there are several possible answers, print any one of them.

It can be shown that, if there is at least one array, then there is an array which satisfies these conditions.

Example

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

Note

Note that the elements of the array that you output don't have to be distinct.

Problem B. Bruteforce

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	512 mebibytes

You are given fixed integers k and w.

For an array a of length n, let us define its *weight* in the following way:

- Let b be the array a sorted in non-descending order.
- The weight of a is then defined as $\sum_{i=1}^{n} \left\lfloor \frac{b_i \cdot i^k}{w} \right\rfloor$.

Here, $\lfloor x \rfloor$ is the largest integer not exceeding x.

For example, if $k = 2$ and $w = 3$, then the weight of $a = [3, 2, 2]$ is equal to:

$$\left\lfloor \frac{2 \cdot 1^2}{3} \right\rfloor + \left\lfloor \frac{2 \cdot 2^2}{3} \right\rfloor + \left\lfloor \frac{3 \cdot 3^2}{3} \right\rfloor = 0 + 2 + 9 = 11.$$

You are given an initial array a, and will be given q queries. Each query changes one element of array a. After each query, you should output the new weight of the array. Since array weights can be really large, you should output them modulo $998\,244\,353$.

Note that the changes persist between queries. For example, the second query is applied to the array which is already changed by the first query.

Input

The first line contains three integers n, k, w ($1 \le n \le 10^5$, $1 \le k \le 5$, $1 \le w \le 5$): the length of the array and the parameters from the statement.

The second line contains n integers a_i ($0 \le a_i \le 10^5$): the elements of the original array.

The third line contains a single integer q ($1 \le q \le 10^5$): the number of queries.

Each of the next q lines contains two integers, pos and x ($1 \le pos \le n$, $0 \le x \le 10^5$). This describes a query that changes a_{pos} into x.

Output

Output q integers: the weights of the array after each change, modulo $998\,244\,353$.

Examples

standard input	standard output
3 1 1 2 2 8 2 2 5 3 6	36 30
4 2 2 1 3 3 7 4 1 1 2 4 3 8 4 8	75 80 103 108

Problem C. Crab's Cannon

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

Mr. Crab is a famous engineer. He has recently invented a new, very powerful weapon called *The Palindromic Cannon*. To make a shot from this mighty cannon, one has to load it with some string consisting of ℓ letters from **unlimited** alphabet.

Let us denote the *palindromic prefix set (PPS)* of a string as the set of all the numbers i ($1 \le i \le \ell$) such that the prefix of length i is a palindrome. For example, the PPS for the string "`abacaba`" is $\{1, 3, 7\}$, and the PPS for the string "`aaaa`" is $\{1, 2, 3, 4\}$. The size of the PPS for the string s is called the *force* of s.

Mr. Crab wanted to test his new weapon. So, he wrote down some string s, ready to be loaded into the cannon. Then he suddenly felt very tired and fell asleep. When he woke up, he couldn't find his string, because enemy spies, Mr. Crabbarc and Mr. Barc, broke into his house and performed some operations with the string.

First, Mr. Crabbarc arrived. He took the string and wrote down its PPS in random order in a sequence. Then, Mr. Barc broke into Mr. Crab's house and might have erased some of the numbers that Mr. Crabbarc wrote.

Mr. Crab needs your help. You need to restore his string s and tell him the force of this string. Since there are many possible ways to restore the string, you need to find the minimum possible force of the restored string.

Input

Input contains muliple test cases. The format of each test case is described below.

The first line of each test case contains two integers n and ℓ ($1 \le n \le 3 \cdot 10^5$, $1 \le \ell \le 10^{18}$), the length of the sequence Mr. Crab found after Mr. Crabbarc and Mr. Barc visited his house, and the length of the initial string.

The second line of the test case contains n integers a_i ($1 \le a_i \le \ell$), the numbers in the sequence. It is guaranteed that all of them are distinct.

After the last test case, there is a line containing "`0 0`". Your program must terminate gracefully after reading this line.

It is guaranteed that the sum of n over all the test cases does not exceed $3 \cdot 10^5$.

Output

For each test case, print line with a single integer: the minimum possible force of the restored string.

Example

standard input	standard output
3 7 1 3 7 4 12 7 1 3 9 3 16 16 1 8 0 0	3 5 3

Note

In the first test case, one of the strings with the minimum possible force is "*abacaba*". Its PPS is $\{1, 3, 7\}$, so its force is 3.

In the second test case, the string could have been "*cbcbcbcbcrab*". Its PPS is $\{1, 3, 5, 7, 9\}$, so its force is 5. One can prove that it is not possible to restore a string with smaller PPS.

In the third test case, consider the string "*crabbarccrabbarc*". Its PPS is $\{1, 8, 16\}$, so its force is 3.

Problem D. Deleting

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

You are given an array $[1, 2, \ldots, n]$, where the number of elements n is even.

In one operation, you can delete two adjacent elements of the array. If these elements are i and j, the cost of this operation is $cost(i, j)$.

In $\frac{n}{2}$ operations, all elements will be deleted. The cost of deleting the whole array is defined as the largest cost among all the $\frac{n}{2}$ operations.

What is the smallest possible cost of deleting the whole array?

Input

The first line of the input contains a single integer n ($2 \le n \le 4000$, n is even).

We are kind today. So we won't provide unnecessary input. It can be shown that it's impossible for two numbers of the same parity to be adjacent at any point, so we won't provide costs for those pairs.

The i-th of the next $n-1$ lines contains $\lfloor\frac{n-i+1}{2}\rfloor$ integers. If i is even, these integers are $cost(i, i+1), cost(i, i+3), \ldots, cost(i, n-1)$. Otherwise, they are $cost(i, i+1), cost(i, i+3), \ldots, cost(i, n)$.

It is guaranteed that the costs form a permutation of numbers from 1 to $(\frac{n}{2})^2$.

Output

Output a single integer: the smallest possible cost of deleting the whole array.

Examples

standard input	standard output
2 1	1
6 2 1 3 4 5 6 7 8 9	6
10 20 21 2 11 25 3 24 18 8 6 17 7 5 22 4 23 14 15 1 19 16 12 10 13 9	14

Note

In the first example, the array is $[1, 2]$, and $cost(1, 2) = 1$. So, the only way to delete the array has the total cost of 1.

In the second example, one of the ways to delete the array is:

- $[1, 2, 3, 4, 5, 6] \to [1, 2, 5, 6]$, deleting pair $(3, 4)$ with cost 6.
- $[1, 2, 5, 6] \to [1, 6]$, deleting pair $(2, 5)$ with cost 5.
- And then deleting pair $(1, 6)$ with cost 3.

The total cost is therefore $\max(6, 5, 3) = 6$.

Problem E. Eulerian?

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

This problem is interactive.

We have hidden from you an undirected graph G on n vertices. It is guaranteed to be connected and to not contain multiple edges or self-loops.

You can ask **up to** 60 **queries** of the following form:

- Consider a subset S of all vertices of G. How many edges are there in the subgraph induced by S? In other words, how many edges in G have both their endpoints in S?

Your goal is to determine whether there exists an Eulerian cycle in this graph. An Eulerian cycle is a path in the graph that goes through every edge exactly once, and it starts and ends in the same vertex.

Note that **graph** G **is fixed before the start of interaction**. In other words, the interactor is not adaptive.

Input

The first line contains a single integer n ($3 \le n \le 10^4$), the number of vertices in G. It is guaranteed that G has no more than 10^5 edges, is connected, and does not contain multiple edges or self-loops.

Interaction Protocol

You start the interaction by reading a line with the integer n.

To find the number of edges in the subgraph of G on k vertices $x_1, x_2, \ldots, x_k$, print a line formatted as "? k x_1 x_2 ... x_k" ($0 \le k \le n$, $1 \le x_i \le n$, all x_i are distinct).

In response, the jury program will print a line with a single integer m: the number of such edges.

In case your query is invalid, or if you asked more than 60 queries, the jury program will print -1 and will finish interaction. You will receive "Wrong answer" outcome. Make sure to terminate your solution immediately to avoid getting other outcomes.

When you have determined whether the graph contains an Eulerian cycle, print a single line: "`! YES`" if such a cycle exists, and "`! NO`" if it doesn't.

After printing each line, do not forget to output the end-of-line and to flush the output. Otherwise, you will receive "Idleness limit exceeded" outcome.

Example

standard input	standard output
3	
	? 2 1 2
1	
	? 2 1 3
0	
	! NO

Note

The hidden graph in the example is the graph with 3 vertices and edges $(2, 1)$ and $(2, 3)$.

Problem F. Fancy Formulas

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

You are given a prime p and a pair of integers (a, b) such that **their sum is not divisible by** p. In one operation, you can do one of the following:

- Replace (a, b) with $(2a \bmod p, (b + p - a) \bmod p)$
- Replace (a, b) with $((a + p - b) \bmod p, 2b \bmod p)$

You have to answer q queries. In the i-th query, find the smallest number of operations needed to transform the pair (a_i, b_i) into the pair (c_i, d_i), or determine that it is impossible.

Note that the order of numbers matters. For example, for $p = 3$, the distance between $(1, 2)$ and $(2, 1)$ is 1, not 0.

Input

The first line contains two integers p and q ($2 \le p \le 10^9 + 7$, p is prime, $1 \le q \le 10^5$): the prime and the number of queries to answer.

The i-th of the next q lines contains four integers a_i, b_i, c_i, d_i ($0 \le a_i, b_i, c_i, d_i < p$, and $a_i + b_i$ **is not divisible by** p).

Output

For each query, if it is impossible to transform (a_i, b_i) into (c_i, d_i), output -1. Otherwise, output the smallest number of operations required to achieve this goal.

Example

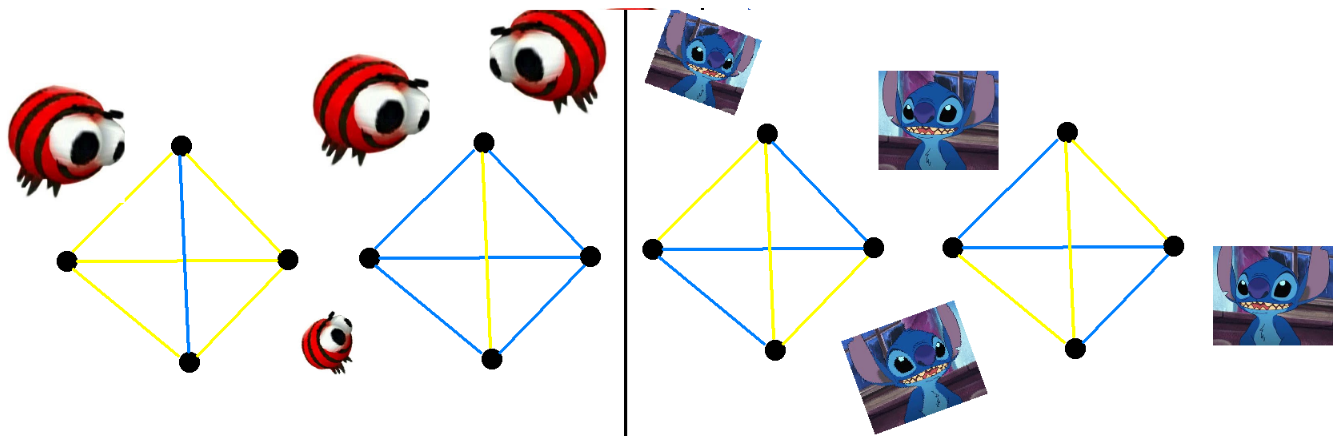
standard input	standard output
5 10 2 1 3 0 2 1 4 4 1 3 4 0 0 2 0 4 3 3 1 2 0 1 0 1 0 3 0 3 0 1 0 1 1 2 4 4 1 0 1 1	2 1 2 -1 -1 0 0 0 1 -1

Problem G. Glory Graph

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

You are given a complete undirected graph on n vertices, each edge is colored blue or yellow. Anton likes a subgraph on 4 vertices if, among its 6 edges, 5 edges have one color, and the 6-th edge has another color. Yahor likes a subgraph on 4 vertices if 3 of its edges are yellow, 3 are blue, and no 3 vertices form a triangle with edges of the same color.

On the image below, on the left, you can see examples of graphs Anton likes. On the right, there are examples of graphs Yahor likes.



Let A be the number of subgraphs Anton likes, and Y be the number of subgraphs Yahor likes. They want to know who likes more subgraphs. To help them, find the value $Y - A$.

Input

The first line of the input contains a single integer n ($4 \le n \le 2000$), the number of vertices in the graph.

The i-th of the next i lines contains a string s_i of length n.

It is guaranteed that:

- For every i from 1 to n, the i-th character of s_i is '`-`'
- For every $i \neq j$, the j-th character of s_i is either '`Y`' or '`B`', where' `Y`' shows that the edge between vertices i and j is yellow, and '`B`' shows that it is blue
- For every $i \neq j$, the j-th character of s_i is equal to the i-th character of s_j

Output

Output a single integer: the value $Y - A$.

Examples

standard input	standard output
5 -YBYB Y-BBB BB-BY YBB-Y BBYY-	2
6 -YYYYY Y-YYBB YY-YYY YYY-YB YBYY-Y YBYBY-	-6

Note

In the first example, Yahor likes subgraphs on vertices $1, 2, 4, 5$ and on vertices $1, 3, 4, 5$. Anton doesn't like any subgraphs there.

Problem H. Hamiltonian

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

You are given a positive integer $K \le 60$. Construct a graph with at most 20 vertices with the following property: there are exactly K unordered pairs of vertices (u, v) such that there is a Hamiltonian path between u and v in this graph.

It can be shown that, under these constraints, the solution always exists.

Recall that a Hamiltonian path is a path between two vertices of a graph that visits each vertex exactly once.

Input

The only line of the input contains a single integer K ($1 \le K \le 60$).

Output

On the first line, output two integers n and m ($2 \le n \le 20$, $0 \le m \le \frac{n(n-1)}{2}$), the number of vertices and the number of edges in your graph respectively.

In each of the next m lines, output two integers u and v ($1 \le u, v \le n$, $u \ne v$), representing the edge (u, v) of your graph. All edges have to be distinct.

Examples

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

Problem I. Intellectual Implementation

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	512 mebibytes

There are n rectangles on the coordinate plane, with sides parallel to the coordinate axis. The i-th rectangle covers all points (x, y) with $l_i \le x \le r_i$ and $d_i \le y \le u_i$.

For simplicity, for every $i \ne j$, we have $l_i \ne l_j$, $r_i \ne r_j$, $l_i \ne r_j$, $d_i \ne d_j$, $u_i \ne u_j$, $d_i \ne u_j$.

Count the number of triples (i, j, k) with $1 \le i < j < k \le n$ for which i-th, j-th, and k-th rectangles are pairwise disjoint (every pair of them has no common points).

Input

The first line of the input contains a single integer n ($1 \le n \le 2 \cdot 10^5$), the number of rectangles.

The i-th of the next n lines contains four integers describing the i-th rectangle: l_i, r_i, d_i, u_i ($-10^9 \le l_i < r_i \le 10^9$, $-10^9 \le d_i < u_i \le 10^9$).

It is guaranteed that, for every $i \ne j$, we have $l_i \ne l_j$, $r_i \ne r_j$, $l_i \ne r_j$, $d_i \ne d_j$, $u_i \ne u_j$, $d_i \ne u_j$.

Output

Output the number of triples (i, j, k) with $1 \le i < j < k \le n$ for which i-th, j-th, and k-th rectangles are pairwise disjoint.

Examples

standard input	standard output
5 1 5 1 5 4 8 2 6 3 7 3 7 2 6 28 32 42 46 42 46	3
6 1 8 6 10 2 5 3 12 3 4 15 20 0 9 2 22 -5 22 -2 23 -7 11 -1 17	0

Problem J. Joke

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

Consider two permutations of integers from 1 to n: p and q. Let us call a binary string s of length n *satisfying* if there exists a matrix a with dimensions $2 \times n$ such that:

- Every integer from 1 to $2n$ appears exactly once in the matrix.
- The elements in the first row are ordered correspondingly to permutation p. More formally, $a_{1,i} < a_{1,j} \iff p_i < p_j$ for $1 \le i < j \le n$.
- The elements in the second row are ordered correspondingly to permutation q. More formally, $a_{2,i} < a_{2,j} \iff q_i < q_j$ for $1 \le i < j \le n$.
- For every i from 1 to n, we have $a_{1,i} < a_{2,i} \iff s_i = 0$.

For two permutations p and q of size n, let us define $f(p, q)$ as the number of satisfying strings s for them.

You are given all elements of p, and several elements of q, but forgot others. Find the sum of $f(p, q)$ over all permutations q with the given known elements, modulo 998 244 353.

Input

The first line of the input contains a single integer n ($1 \le n \le 100$).

The second line of the input contains n integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$, all p_i are distinct), a permutation of numbers from 1 to n.

The second line of the input contains n integers $q_1, q_2, \ldots, q_n$ ($0 \le q_i \le n$, $q_i \ne q_j$ when $q_i \ne 0$ and $q_j \ne 0$). If $q_i \ne 0$, the respective element is given. If $q_i = 0$, its value is forgotten. All given elements are distinct.

Output

Output the sum of $f(p, q)$ over all valid permutations q modulo 998 244 353.

Examples

standard input	standard output
2 1 2 2 1	3
4 4 3 2 1 4 3 2 1	16
5 1 2 3 4 5 0 0 0 0 0	1546
6 1 6 2 5 3 4 0 1 0 2 0 3	52

Problem K. K-onstruction

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

You are given an integer K such that $1 \le K \le 10^6$. Construct any array A of numbers for which the following properties hold:

- The size of A is between 1 and 30;
- All elements are integers between -10^{16} and 10^{16};
- Let N be the size of A. Then there are exactly K subsets S (possibly empty) of set $\{1, 2, \ldots, N\}$ for which $\sum_{i \in S} A_i = 0$.

It can be shown that, under the constraints above, such array A always exists.

Input

The first line contains a single integer t ($1 \le t \le 1000$), the number of test cases.

Each of the next t lines contains a single integer K ($1 \le K \le 10^6$).

Output

For each test case, on the first line, output a single integer N ($1 \le N \le 30$), the size of your array.

On the second line, output N integers $A_1, A_2, \ldots, A_N$ ($-10^{16} \le A_i \le 10^{16}$), the elements of the array.

Example

standard input	standard output
2 3 16	5 2021 -1000 -1021 -2000 -21 4 0 0 0 0

Note

Note that the elements of the array don't have to be distinct.

Problem L. Little LCS

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

A string consisting of letters '`A`', '`B`', '`C`' is *good* if every two adjacent letters are different.

A pair of two good strings (s, t) of length $2n + 1$ is *awesome* if the length of their longest common subsequence is exactly n.

You are given two strings, s and t, consisting of letters '`A`', '`B`', '`C`' and question marks ('`?`'). Find the number of ways to replace each '`?`' with one of '`A`', '`B`', '`C`', so that the pair (s, t) is awesome, modulo 998 244 353.

Input

The first line contains a single integer t ($1 \le t \le 10^5$), the number of test cases.

The first line of each test case contains a single integer n ($1 \le n \le 10^6$).

The second line contains a string s of length $2n + 1$ consisting of characters '`A`', '`B`', '`C`', '`?`'.

The third line contains a string t of length $2n + 1$ consisting of characters '`A`', '`B`', '`C`', '`?`'.

It is guaranteed that the sum of n over all test cases does not exceed 10^6.

Output

For each test case, output the number of ways to replace each '`?`' with one of '`A`', '`B`', '`C`' so that the pair (s, t) is awesome, modulo 998 244 353.

Example

standard input	standard output
5 1 ABA CBC 1 A?A C?C 1 ??? ??? 2 AA??? ????B 3 ?A?B?A? ???????	1 3 24 0 2

Note

In the first test case, pair $(\texttt{ABA}, \texttt{CBC})$ is awesome.

In the second test case, there are 3 ways to replace question marks to get an awesome pair: $(\texttt{ABA}, \texttt{CBC})$, $(\texttt{ACA}, \texttt{CBC})$, $(\texttt{ABA}, \texttt{CAC})$.

Problem M. Math

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

You are given an array a of n **distinct** positive integers. Find the number of pairs (i, j) with $1 \leq i, j \leq n$ for which the number $a_i^2 + a_j$ is a square of an integer.

Input

The first line of the input contains a single integer n $(1 \leq n \leq 10^6)$, the size of the array.

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

Output

Output a single integer: the answer to the problem.

Example

standard input	standard output
5 1 2 3 4 5	2

Note

In the example, there are two such pairs, corresponding to $1^2 + 3 = 4 = 2^2$ and $2^2 + 5 = 9 = 3^2$.