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 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 x1 x2 ... xk`" ($0 \le k \le n$, $1 \le x_i \le n$, 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 1 0	 ? 2 1 2 ? 2 1 3 ! 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

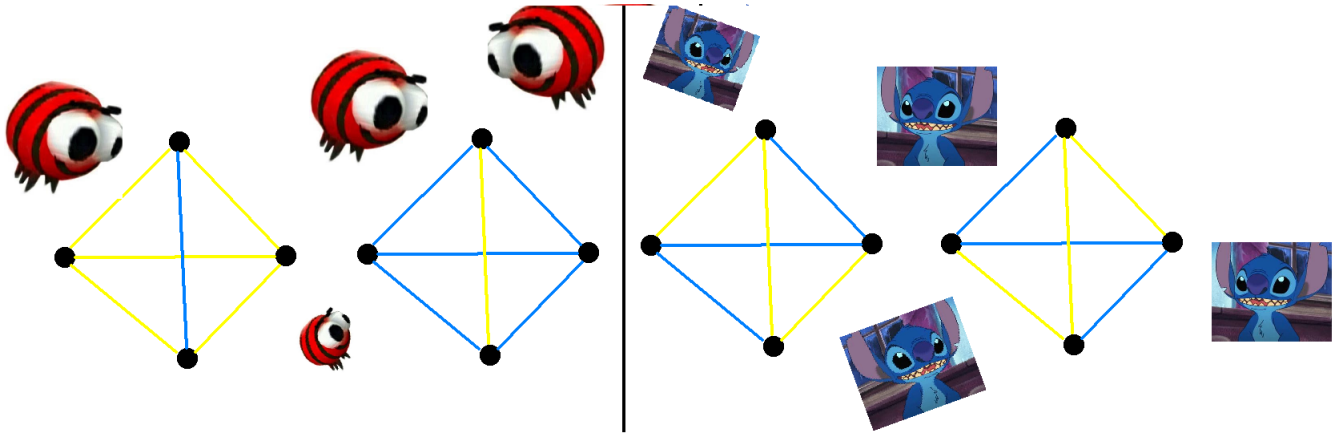
standard input	standard output
5 10 2 1 3 0 2 1 4 4 1 3 4 0 0 2 0 4 3 3 1 2 0 1 0 1 0 3 0 3 0 1 0 1 1 2 4 4 1 0 1 1	2 1 2 -1 -1 0 0 0 1 -1

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 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 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 $1 \le i, j \le n$ 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 ($1 \le n \le 10^6$), the size of the array.

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

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 $2^2 + 5 = 9 = 3^2$.

Problem N. New Companies

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

The CEO of the company named S is planning to merge with another company named T. The CEO wants to keep the original company name S through the merge process on the plea that both company names are mixed into a new one.

The CEO insists that the mixed company name is produced as follows.

Let s be an arbitrary subsequence of S, and t be an arbitrary subsequence of T. The new company name must be a string of the same length to S obtained by alternatively lining up the characters in s and t. More formally, $s_0 + t_0 + s_1 + t_1 + \ldots$ or $t_0 + s_0 + t_1 + s_1 + \ldots$ can be used as the company name after merging. Here,s_k denotes the k-th (0-based) character of string s. Please note that the lengths of s and t will be different if the length of S is odd. In this case, the company name after merging is obtained by $s_0 + t_0 + \ldots + t_{|S|/2} + s_{|S|/2+1}$ or $t_0 + s_0 + \ldots + s_{|S|/2} + t_{|S|/2+1}$ ($|S|$ denotes the length of S and "/" denotes integer division).

A subsequence of a string is a string which is obtained by erasing zero or more characters from the original string. For example, the strings "abe", "abcde" and "" (the empty string) are all subsequences of the string "abcde".

You are a programmer employed by the acquiring company. You are assigned a task to write a program that determines whether it is possible to make S, which is the original company name, by mixing the two company names.

Input

The first line contains a string S which denotes the name of the company that you belong to. The second line contains a string T which denotes the name of the target company of the planned merging. The two names S and T are non-empty and of the same length no longer than 10^3 characters, and all the characters used in the names are lowercase English letters.

Output

Print "Yes" in a line if it is possible to make original company name by combining S and T. Otherwise, print "No" in a line.

Examples

standard input	standard output
acmicpc tsukuba	No
hoge moen	Yes
abcdefg xacxegx	Yes

Problem O. Operations With Password

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

Password authentication is used in a lot of facilities. The office of JAG also uses password authentication. A password is required to enter their office. A password is a string of N digits '0'-'9'. This password is changed on a regular basis. Taro, a staff of the security division of JAG, decided to use the following rules to generate a new password from an old one.

1. The new password consists of the same number N of digits to the original one and each digit appears at most once in the new password. It can have a leading zero. (Note that an old password may contain same digits twice or more.)
2. The new password maximizes the difference from the old password within constraints described above. (Definition of the difference between two passwords is described below.)
3. If there are two or more candidates, the one which has the minimum value when it is read as an integer will be selected.

The difference between two passwords is defined by $min(|a - b|, 10^N - |a - b|)$, where a and b are the integers represented by the two passwords. For example, the difference between "11" and "42" is 31, and the difference between "987" and "012" is 25.

Taro would like to use a computer to calculate a new password correctly, but he is not good at programming. Therefore, he asked you to write a program. Your task is to write a program that generates a new password from an old password.

Input

The input consists of a single test case. The first line of the input contains a string S which denotes the old password. You can assume that the length of S is no less than 1 and no greater than 10. Note that old password S may contain same digits twice or more, and may have leading zeros.

Output

Print the new password in a line.

Examples

standard input	standard output
201	701
512	012
99999	49876
765876346	265874931

Problem P. Panels

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

You are in front of a linear gimmick of a game. It consists of N panels in a row, and each of them displays a right or a left arrow.

You can step in this gimmick from any panel. Once you get on a panel, you are forced to move following the direction of the arrow shown on the panel and the panel will be removed immediately. You keep moving in the same direction until you get on another panel, and if you reach a panel, you turn in (or keep) the direction of the arrow on the panel. The panels you passed are also removed. You repeat this procedure, and when there is no panel in your current direction, you get out of the gimmick.

For example, when the gimmick is the following image

and you first get on the 2nd panel from the left, your moves are as follows.

- Move right and remove the 2nd panel.
- Move left and remove the 3rd panel.
- Move right and remove the 1st panel.
- Move right and remove the 4th panel.
- Move left and remove the 5th panel.
- Get out of the gimmick.

You are given a gimmick with N panels. Compute the maximum number of removed panels after you get out of the gimmick.

Input

The input consists of two lines. The first line contains an integer N ($1 \le N \le 10^5$) which represents the number of the panels in the gimmick. The second line contains a character string S of length N, which consists of '>' or '<'. The i-th character of S corresponds to the direction of the arrow on the i-th panel from the left. '<' and '>' denote left and right directions respectively.

Output

Output the maximum number of removed panels after you get out of the gimmick.

Example

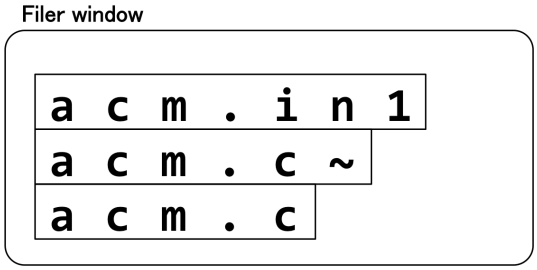
standard input	standard output
7 >><>><<	7
5 >><<<	5
6 ><<>><	6
7 <<><<>>	5

Problem Q. Quick Deletion

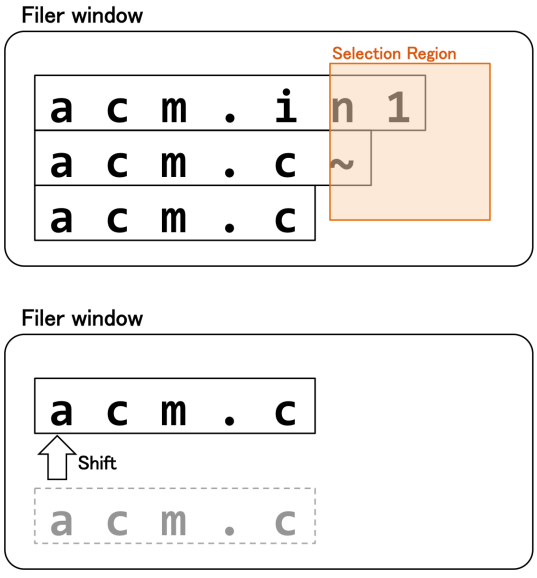
Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

You are using an operating system named "Jaguntu". Jaguntu provides "Filer", a file manager with a graphical user interface.

When you open a folder with Filer, the name list of files in the folder is displayed on a Filer window. Each filename is displayed within a rectangular region, and this region is called a filename region. Each filename region is aligned to the left side of the Filer window. The height of each filename region is 1, and the width of each filename region is the filename length. For example, when three files "`acm.in1`", "`acm.c `", and "`acm.c`" are stored in this order in a folder, it looks like this on the Filer window.



You can delete files by taking the following steps. First, you select a rectangular region with dragging a mouse. This region is called selection region. Next, you press the delete key on your keyboard. A file is deleted if and only if its filename region intersects with the selection region. After the deletion, Filer shifts each filename region to the upside on the Filer window not to leave any top margin on any remaining filename region. For example, if you select a region from the first picture below, then the two files "`acm.in1`" and "`acm.c `" are deleted, and the remaining file "`acm.c`" is displayed on the top of the Filer window as at the second picture.



You are opening a folder storing N files with Filer. Since you have almost run out of disk space, you want to delete unnecessary files in the folder. Your task is to write a program that calculates the minimum number of times to perform deletion operation described above.

Input

The input consists of a single test case.

The first line contains one integer N ($1 \le N \le 1,000$), which is the number of files in a folder. Each of the next N lines contains a character D_i and an integer L_i: D_i indicates whether the i-th file should be deleted or not, and

L_i ($1 \le L_i \le 1,000$) is the filename length of the i-th file. If D_i is '`y`', the i-th file should be deleted. Otherwise, D_i is always '`n`', and you should not delete the i-th file.

Output

Output the minimum number of deletion operations to delete only all the unnecessary files.

Examples

standard input	standard output
3 y 7 y 6 n 5	1
3 y 7 n 6 y 5	2
6 y 4 n 5 y 4 y 6 n 3 y 6	2