

Problem A. Adjacent Decrements

Input file: *standard input*
Output file: *standard output*
Time limit: 7 seconds
Memory limit: 1024 mebibytes

Consider the following operation on an array a of length n . Choose an index i ($1 \leq i < n$), and then:

- if $a_i = a_{i+1}$, decrease both a_i and a_{i+1} by 1;
- otherwise, decrease the smaller of a_i and a_{i+1} by 1.

Every element of the array must be non-negative at all times, so an operation may only be applied when it does not make any element negative.

An array a is called *good* if it is possible to make all of its elements equal to 0 by applying this operation any number of times, possibly zero.

You are given n and k . Count the good arrays a of length n satisfying $0 \leq a_i \leq k$ for every i , modulo 998 244 353.

Input

The only line contains two integers n and k ($1 \leq n, k \leq 250\,000$).

Output

Print a single integer: the number of good arrays, taken modulo 998 244 353.

Examples

<i>standard input</i>	<i>standard output</i>
5 2	36
7 3	855

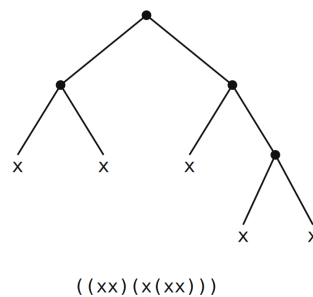
Problem B. Binary Rotations

Input file: *standard input*
 Output file: *standard output*
 Time limit: 7 seconds
 Memory limit: 1024 mebibytes

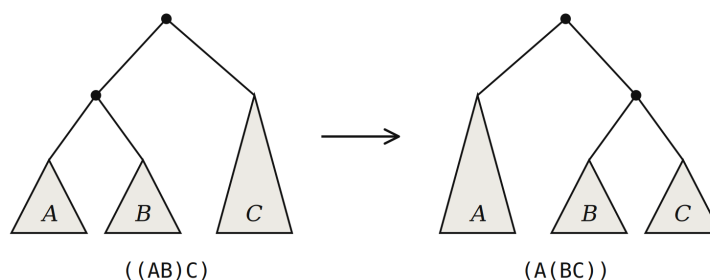
A *valid expression* is defined recursively:

- x is a valid expression;
- if A and B are valid expressions, then (AB) is also a valid expression.

For example, $((xx)(x(xx)))$ is a valid expression with five occurrences of x . Every valid expression corresponds to a binary tree: each x is a leaf, and each pair of brackets is an internal node whose children are the two subexpressions it encloses.



In one operation, you may replace a subexpression of the form $((AB)C)$ with $(A(BC))$, where A , B , and C are arbitrary valid expressions. In the tree representation, this rotates a single edge.



We say that an expression u can be obtained from an expression v if u can be produced from v by applying zero or more operations. In particular, every expression can be obtained from itself.

You are given an integer ℓ . The number of valid expressions with exactly ℓ occurrences of x is the Catalan number

$$n = C_{\ell-1} = \frac{1}{\ell} \binom{2\ell-2}{\ell-1},$$

and all of them are enumerated in lexicographical order as $S_1, S_2, \dots, S_n$, using the character order ' $(< x <)$ '. Here, $\binom{a}{b} = \frac{a!}{b!(a-b)!}$ is the binomial coefficient.

You are also given a permutation $p_1, p_2, \dots, p_n$ of the integers from 1 to n ; the value p_i represents the expression S_{p_i} . Count the pairs (i, j) with $1 \leq i < j \leq n$ such that S_{p_i} can be obtained from S_{p_j} .

Input

The first line contains one integer ℓ ($1 \leq \ell \leq 14$).

The second line contains $n = C_{\ell-1}$ integers $p_1, p_2, \dots, p_n$: a permutation of $1, 2, \dots, n$.

Output

Print one integer: the number of pairs (i, j) such that $1 \leq i < j \leq n$ and S_{p_i} can be obtained from S_{p_j} .

Example

<i>standard input</i>	<i>standard output</i>
4 3 1 5 2 4	3

Note

For $\ell = 4$ there are $n = C_3 = 5$ valid expressions, in lexicographical order: $S_1 = ((\mathbf{x}\mathbf{x})\mathbf{x})\mathbf{x}$, $S_2 = ((\mathbf{x}(\mathbf{x}\mathbf{x}))\mathbf{x})$, $S_3 = ((\mathbf{x}\mathbf{x})(\mathbf{x}\mathbf{x}))$, $S_4 = (\mathbf{x}((\mathbf{x}\mathbf{x})\mathbf{x}))$, $S_5 = (\mathbf{x}(\mathbf{x}(\mathbf{x}\mathbf{x})))$.

A single operation turns S_1 into S_2 or into S_3 , turns S_2 into S_4 , and turns each of S_3 and S_4 into S_5 ; no operation applies to S_5 .

The three pairs that count are $(i, j) = (1, 2)$, since S_3 can be obtained from S_1 ; $(3, 4)$, since S_5 can be obtained from S_2 ; and $(3, 5)$, since S_5 can be obtained from S_4 .

Problem C. Circle Construction

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 1024 mebibytes

You are given an integer r .

Construct six pairwise distinct points $P_1, P_2, \dots, P_6$ with integer coordinates such that all of the following conditions hold:

- P_1, P_2, P_3 , and P_4 lie on one circle of radius exactly r ;
- P_3, P_4, P_5 , and P_6 lie on one circle;
- P_1, P_2, P_5 , and P_6 lie on one circle;
- the three circles above are pairwise distinct.

The centers of all three circles do not have to have integer coordinates. The radii of the second and third circles do not have to be integer either. Only the coordinates of the six constructed points are required to be integer.

Input

The first line contains a single integer t ($1 \leq t \leq 50$): the number of test cases.

Each of the next t lines contains one integer r ($1 \leq r \leq 150$): the required radius of the first circle.

Output

For each test case, print six lines. The i -th line must contain two integers x_i and y_i ($|x_i|, |y_i| \leq 10\,000$), denoting the coordinates of P_i .

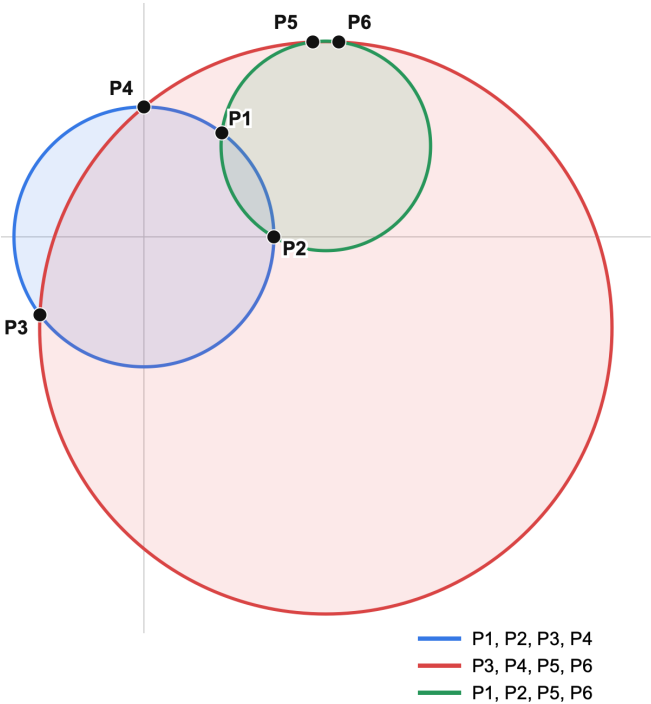
If there are several valid constructions, print any one of them.

Example

<i>standard input</i>	<i>standard output</i>
1	6 8
10	10 0
	-8 -6
	0 10
	13 15
	15 15

Note

The example shows one possible construction. Other valid constructions are accepted as well.



Problem D. Dynamic Adjustment

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 1024 mebibytes

For an array $a_1, a_2, \dots, a_m$ of integers, let $f(a_1, a_2, \dots, a_m)$ be the smallest possible value of

$$\sum_{i=1}^m |a_i - b_i|$$

over all integer arrays $b_1, b_2, \dots, b_m$ with $b_1 \leq b_2 \leq \dots \leq b_m$. The values b_i may be arbitrary integers.

You are given an array $x_1, x_2, \dots, x_n$ and a permutation $p_1, p_2, \dots, p_n$ of the integers from 1 to n .

For each i from 1 to n , consider the indices $p_1, p_2, \dots, p_i$ sorted in increasing order, say $q_1 < q_2 < \dots < q_i$, and let a be the array $x_{q_1}, x_{q_2}, \dots, x_{q_i}$ of length i . In other words, a consists of the elements of x at the first i positions of the permutation, taken in the order in which they occur in x . Compute $f(a)$ for every i .

Input

The first line contains a single integer n ($1 \leq n \leq 200\,000$): the length of the array.

The second line contains n integers $x_1, x_2, \dots, x_n$ ($-10^9 \leq x_i \leq 10^9$).

The third line contains n distinct integers $p_1, p_2, \dots, p_n$ ($1 \leq p_i \leq n$).

Output

Print n integers separated by spaces: the i -th of them must be the value of $f(a)$ for the array a built from $p_1, p_2, \dots, p_i$.

Examples

<i>standard input</i>	<i>standard output</i>
5 5 4 3 2 1 1 2 3 4 5	0 1 2 4 6
3 3 2 1 1 3 2	0 2 2

Note

Arrays in the first example:

- $[5], [5, 4], [5, 4, 3], [5, 4, 3, 2], [5, 4, 3, 2, 1]$

Arrays in the second example:

- $[3], [3, 1], [3, 2, 1]$

For $[3, 1]$, changing both values to 1 (or both to 2, or both to 3) costs 2. After the middle element is inserted, changing $[3, 2, 1]$ to $[2, 2, 2]$ also costs 2.

Problem E. Equal Totients

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 1024 mebibytes

Let $\phi(n)$ denote Euler's totient function: the number of integers in $[1, n]$ that are coprime to n .

You are given two integers ℓ and r with $r \geq \frac{5\ell}{3}$. Find two distinct integers x and y with $\ell \leq x, y \leq r$ such that $\phi(x) = \phi(y)$, or report that no such pair exists.

Input

The first line contains a single integer t ($1 \leq t \leq 20$): the number of test cases.

Each of the next t lines contains two integers ℓ and r ($1 \leq \ell < r \leq 10^{17}$) describing one test case. It is guaranteed that $r \geq \frac{5\ell}{3}$.

Output

For each test case, print two integers x and y ($\ell \leq x, y \leq r$, $x \neq y$) such that $\phi(x) = \phi(y)$, or “-1 -1” if no such pair exists. If there are multiple valid pairs, print any one of them.

Example

<i>standard input</i>	<i>standard output</i>
5	1 2
1 3	5 8
5 10	15 16
15 30	21 26
21 50	52 56
50 100	

Problem F. Flipping Lamps

Input file: *standard input*
Output file: *standard output*
Time limit: 7 seconds
Memory limit: 1024 mebibytes

There are n lamps in a row, all initially off. The show lasts for m days.

On day i , you must choose exactly a_i distinct lamps and toggle each chosen lamp: a lamp that is off becomes on, and a lamp that is on becomes off.

A *plan* is the sequence of chosen sets over all m days. Two plans are different if there is a day on which the chosen sets differ.

Count the plans that leave exactly r lamps on after day m , modulo 998 244 353.

Input

The first line contains three integers n , m , and r ($1 \leq n \leq 10\,000$, $1 \leq m \leq 200\,000$, $0 \leq r \leq n$).

The second line contains m integers $a_1, a_2, \dots, a_m$ ($0 \leq a_i \leq n$).

Output

Print a single integer: the number of plans leaving exactly r lamps on after day m , taken modulo 998 244 353.

Examples

<i>standard input</i>	<i>standard output</i>
3 2 1 1 2	6
7 4 4 1 2 4 5	57960

Problem G. Game

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 1024 mebibytes

There are n piles of stones; the i -th pile contains a_i stones.

Two players take turns making moves. In one move, the current player chooses a pile and removes a positive number of stones k from it. The move is allowed only if the binary representation of k contains at most m ones.

The player who cannot make a move loses. Determine who wins if both players play optimally.

Input

The first line contains a single integer t ($1 \leq t \leq 10^4$): the number of test cases.

Each test case is described by two lines. The first line contains two integers n and m ($1 \leq n \leq 2 \cdot 10^5$, $1 \leq m \leq 60$): the number of piles and the maximum number of ones allowed in the binary representation of a removed amount.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq 10^{18}$): the sizes of the piles. The sizes are not necessarily distinct.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print **First** if the player who moves first wins, and **Second** otherwise.

Example

<i>standard input</i>	<i>standard output</i>
3	Second
1 2	First
7	Second
2 1	
5 6	
3 2	
7 14 21	

Note

In the first test case, $m = 2$, and the only pile contains 7 stones. The first player cannot take all 7 stones, because $7 = 111_2$ contains three ones. It can be checked that every allowed move leads to a position in which the second player wins.

In the second test case, the first player can take 2 stones from the first pile, leaving piles of sizes 3 and 6.

Problem H. Hamiltonian LCM

Input file: *standard input*
Output file: *standard output*
Time limit: 7 seconds
Memory limit: 1024 mebibytes

You are given an array a of n positive integers.

Rearrange its elements in any order. If the resulting array is $b_1, b_2, \dots, b_n$, its *value* is

$$\text{lcm}(b_1, b_2) + \text{lcm}(b_2, b_3) + \dots + \text{lcm}(b_{n-1}, b_n),$$

the sum of the least common multiples of all pairs of adjacent elements.

Find the maximum value over all rearrangements of a .

Input

The first line contains a single integer t ($1 \leq t \leq 10$): the number of test cases.

The first line of each test case contains a single integer n ($1 \leq n \leq 100\,000$): the number of elements in the array.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq 7$).

It is guaranteed that the sum of n over all test cases does not exceed 100 000.

Output

For each test case, print a single integer: the maximum possible value.

Examples

<i>standard input</i>	<i>standard output</i>
1 2 5 2	10
1 6 2 2 2 3 3 3	30

Problem I. Interval MST

Input file: *standard input*
Output file: *standard output*
Time limit: 7 seconds
Memory limit: 1024 mebibytes

You are given a sequence of integers $x_1, x_2, \dots, x_n$. Consider the weighted undirected graph G on n vertices, numbered 1 through n , that contains, for every pair of vertices u and v with $u < v$, the edge $\{u, v\}$ of weight $x_v - x_u$.

Note that x is not necessarily sorted, so edge weights may be negative.

For a pair (ℓ, r) with $\ell \leq r$, let $G[\ell, r]$ denote the subgraph of G induced by the vertices $\ell, \ell+1, \dots, r$; that is, the graph on those vertices keeping exactly the edges of G with both endpoints in that range. Define $f(\ell, r)$ as the minimum possible total weight of a spanning tree of $G[\ell, r]$. In particular, $f(\ell, \ell) = 0$.

You are given q pairs (ℓ, r) . Compute $f(\ell, r)$ for each of them.

Input

The first line contains two integers n and q ($1 \leq n, q \leq 2 \cdot 10^5$): the length of the sequence and the number of queries.

The second line contains n integers $x_1, x_2, \dots, x_n$ ($-10^9 \leq x_i \leq 10^9$).

Each of the next q lines contains two integers ℓ and r ($1 \leq \ell \leq r \leq n$) describing one query.

Output

Print q lines. The i -th line must contain a single integer: the answer to the i -th query.

Example

<i>standard input</i>	<i>standard output</i>
3 4	-10
0 10 0	10
1 3	-10
1 2	0
2 3	
2 2	

Note

For the first query, the three edge weights are 10, 0, and -10 . Choosing the edges of weights 0 and -10 gives a spanning tree of total weight -10 .

The second and third queries contain two vertices, so their answers are the weights of their only edges: 10 and -10 , respectively. The final query contains one vertex and therefore has answer 0.

Problem J. Jump Cycle

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 1024 mebibytes

You are given a positive even integer n . Consider an $n \times n$ grid whose rows and columns are both numbered from 1 to n . The cell in row r and column c is denoted (r, c) .

Your task is to arrange all n^2 cells of the grid into a sequence $p_1, p_2, \dots, p_{n^2}$ in which every cell appears exactly once such that, with the convention $p_{n^2+1} = p_1$, for every i from 1 to n^2 , the cells p_i and p_{i+1} lie in the same row or in the same column and satisfy

$$|r_i - r_{i+1}| + |c_i - c_{i+1}| = ((i - 1) \bmod 4) + 1,$$

where $p_i = (r_i, c_i)$ and $p_{i+1} = (r_{i+1}, c_{i+1})$.

In other words, the sequence should be a closed tour of length n^2 visiting every cell once, each move is horizontal or vertical, and the move lengths form a sequence $1, 2, 3, 4, 1, 2, 3, 4, \dots$ repeating cyclically, including the final move from p_{n^2} back to p_1 .

If such a sequence exists, output any one of them.

Input

The first line contains a single even integer n ($4 \leq n \leq 200$).

Output

If no valid sequence exists, print a single line containing NO.

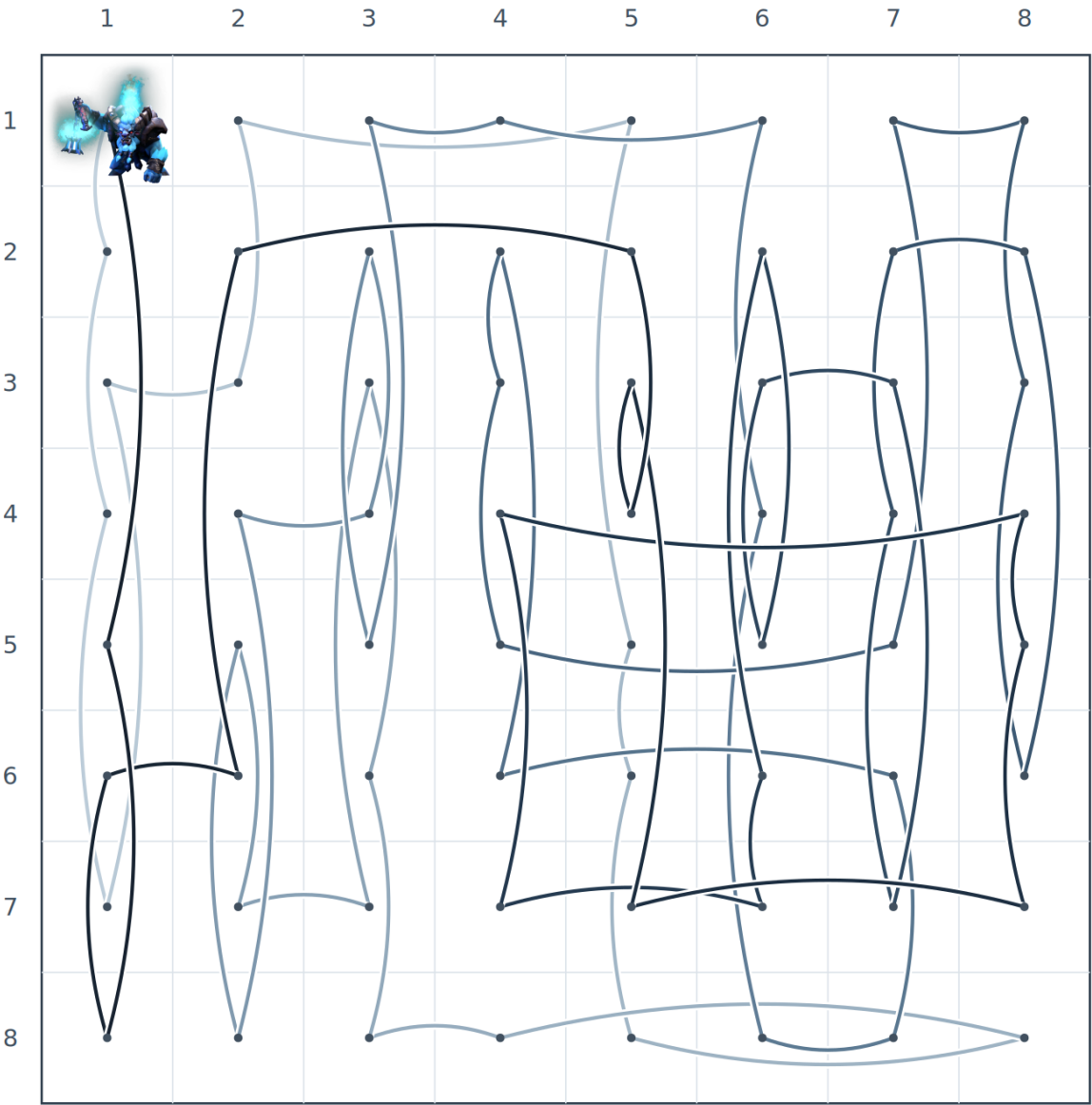
Otherwise, print YES on the first line, and then print n lines containing n integers each. The j -th column of the i -th row should contain one integer k such that, in your answer, p_k is the cell (i, j) .

If there are multiple valid answers, print any one of them.

Example

<i>standard input</i>	<i>standard output</i>
8	YES 1 7 25 26 8 27 37 38 2 60 23 33 59 48 42 41 5 6 16 34 57 46 45 39 3 21 22 52 58 28 43 53 64 19 24 35 9 47 36 54 62 61 15 32 10 49 31 40 4 18 17 51 56 50 44 55 63 20 14 13 11 29 30 12

Note



move 1 (light) move 64, back to the start (dark)

Problem K. Keep Colors Apart

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 1024 mebibytes

There are n events. Event i has a deadline d_i , a color c_i , and a weight w_i . Every color occurs at most twice among the events.

You may choose any subset of the events and assign each chosen event to a positive integer day, so that all of the following hold:

- no two chosen events are assigned to the same day;
- if event i is assigned to day x , then $x \leq d_i$;
- if two consecutive days x and $x + 1$ both have an event assigned to them, those two events have different colors.

Days with no event assigned to them are allowed, and there is no limit on how large a day number may be. You may also choose no events at all, in which case the total weight is 0.

Find the maximum possible total weight of the chosen events.

Input

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

The first line of each test case contains a single integer n ($1 \leq n \leq 200\,000$): the number of events.

Each of the next n lines contains three integers d_i , c_i , and w_i ($1 \leq d_i, c_i, w_i \leq 10^9$): the deadline, the color, and the weight of the i -th event.

For every test case, each value occurs at most twice among $c_1, c_2, \dots, c_n$. It is guaranteed that the sum of n over all test cases does not exceed 200 000.

Output

For each test case, print a single integer: the maximum possible total weight.

Example

<i>standard input</i>	<i>standard output</i>
5	100
2	20
1 1 100	74
2 1 99	132
2	165
2 1 16	
3 1 4	
4	
2 1 37	
1 2 37	
2 2 12	
1 1 60	
7	
4 3 13	
2 3 3	
1 1 28	
6 2 30	
4 1 16	
5 2 21	
3 4 37	
10	
6 3 27	
5 4 8	
3 5 27	
2 2 11	
1 5 6	
1 1 33	
6 3 28	
1 4 32	
6 2 21	
2 1 30	

Note

In the first test case, the two events have the same color. Their deadlines do not leave an empty day between them, so at most one can be chosen; event 1 has the larger weight.

In the second test case, assign the first event to day 1 and the second event to day 3. The empty day 2 makes it possible to choose both events even though their colors are equal.

In the third test case, choose events 1 and 2 and assign them to days 2 and 1, respectively, for a total weight of 74.

In the fourth test case, one optimal schedule uses events 3, 4, 7, 5, and 6 on days 1 through 5, in that order. Its total weight is $28 + 30 + 37 + 16 + 21 = 132$. In particular, both occurrences of colors 1 and 2 are used, but equal colors are never placed on consecutive days.

In the fifth test case, one optimal schedule uses events 8, 10, 3, 1, 9, and 7 on days 1 through 6. Its total weight is 165.

Problem L. Leaf Meeting

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 1024 mebibytes

You are given an undirected tree with n vertices. A *leaf* is a vertex of degree 1. Let k be the number of leaves; initially, there is one token on each leaf, so there are k tokens in total.

For every token, you must choose an infinite sequence of leaves $a_1, a_2, a_3, \dots$ with $a_i \neq a_{i+1}$ for all $i \geq 1$, and with $a_1 \neq v$, where v is the leaf the token starts on. The sequences for different tokens can be chosen independently. The chosen sequence determines the route of the token: it first walks along the shortest path from v to a_1 , then along the shortest path from a_1 to a_2 , then from a_2 to a_3 , and so on indefinitely.

In one move, every token simultaneously traverses exactly one edge of its own route. No token may stay in place.

Determine whether the sequences can be chosen so that, after some finite number of moves, all k tokens are on the same leaf at the same time.

Input

The first line contains a single integer t ($1 \leq t \leq 10^4$): the number of test cases.

The first line of each test case contains an integer n ($2 \leq n \leq 2 \cdot 10^5$): the number of vertices in the tree.

Each of the next $n - 1$ lines contains two integers u and v ($1 \leq u, v \leq n$): a bidirectional edge between vertices u and v . It is guaranteed that these edges form a tree.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print the answer on a separate line. If it is possible to choose the required sequences for all tokens so that they all gather in one leaf at the same time, print **YES**, otherwise print **NO**.

Example

<i>standard input</i>	<i>standard output</i>
5	NO
2	NO
1 2	NO
3	NO
2 1	YES
2 3	
4	
1 2	
1 3	
3 4	
6	
1 2	
2 3	
2 4	
4 5	
4 6	
8	
1 2	
1 3	
1 4	
1 5	
1 6	
1 7	
1 8	

Problem M. Matching Substrings

Input file: *standard input*
Output file: *standard output*
Time limit: 4 seconds
Memory limit: 1024 mebibytes

You are given two strings a and b .

String s is a *substring* of string t if s can be obtained from t by deleting some (possibly zero) characters from the beginning and some (possibly zero) characters from the end. In particular, the empty string is a substring of every string.

You are given q query strings. For a query string c of length m and for every k from 1 to m , count the distinct strings x such that

- x is a substring of a ;
- $xc_1c_2 \dots c_k$ is a substring of b .

The string x may be empty. Two strings x are considered the same if they are equal as strings, no matter where they occur in a .

Input

The first line contains a single integer t ($1 \leq t \leq 10^4$): the number of test cases.

The first line of each test case contains the string a ($1 \leq |a| \leq 150\,000$).

The second line contains the string b ($1 \leq |b| \leq 150\,000$).

The third line contains a single integer q ($1 \leq q \leq 150\,000$): the number of queries.

Each of the next q lines contains one query string c ($1 \leq |c| \leq 150\,000$).

All strings consist of lowercase English letters.

It is guaranteed that the sum of $|a|$ over all test cases does not exceed 150 000, the sum of $|b|$ over all test cases does not exceed 150 000, and the sum of $|c|$ over all query strings of all test cases does not exceed 150 000.

Output

For each query, print $|c|$ integers: the k -th of them must be the number of distinct strings x that are substrings of a and for which $xc_1c_2 \dots c_k$ is a substring of b .

Examples

<i>standard input</i>	<i>standard output</i>
4 ab cababa	3 2 3 3
5 a b ab ba c aaa aaaaa	2 2 1 4 4 4 4 4 3 4 4 3 2 0 1 1
4 a aa aaa aaaa abc xyz 4 a xy z abc banana anaban	1 0 0 0 4 3 2 4 2 4 4 4 3
5 a na an ban n	
1 ab aaba 1 ab	4 2

Note

In the second example, for the prefix **a**, the four valid strings x are the empty string, **a**, **b**, and **ab**. In particular, **a** and **b** must both be counted even though they have the same length. For the full query string **ab**, only the empty string and **a** are valid.