Problem A. Permutations and Cycles (Minimum Version)

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

For a given x, a permutation of size n is called *good* if for each $1 \le i < n$ the condition $p_i + p_{i+1} \le x$ holds. Find any good permutation with the **minimum** number of cycles.

A permutation of size n is a sequence of n distinct integers from 1 to n.

A cycle of a permutation p is a sequence of indices $i_1, i_2, \ldots, i_k$ such that $p_{i_1} = i_2$, $p_{i_2} = i_3$, ..., $p_{i_k} = i_1$. The cycles obtained by a cyclic shifting of the sequence are considered to be the same.

Input

The first line contains an integer t ($1 \le t \le 2 \cdot 10^5$), the number of test cases. The test cases follow.

Each test case is given on a line with two integers n ($2 \le n \le 2 \cdot 10^5$) and x ($n+1 \le x \le 2 \cdot n - 1$). These constraints guarantee that at least one *good* permutation exists.

The sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print two lines. The first one should contain the minimum number of cycles in a good permutation of length n. The second line should consist of n integers: the permutation itself. If multiple such permutations exist, print any one of them.

Example

standard input	*standard output*
2 2 3 6 10	1 2 1 1 5 4 6 3 2 1

Problem B. Segments Removal

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

Consider points $1, 2, \ldots, x$ on a coordinate axis. You are given a collection of segments that start and end in these points. Each segment has weight and penalty associated with it.

Initially, you have a score of 0. You can make moves. In each move, you select a segment from the given collection, remove it from the collection, and your score decreases by the penalty associated with the segment. In return, your score increases by the weight of the segment multiplied by the number of points with integer coordinates such that, at this moment of time, this segment is the **only** segment in the collection that covers these points. A segment is considered to cover its endpoints.

The total score is the sum of scores for the moves you make. Find the maximum total score you can achieve.

Input

The first line contains an integer t ($1 \le t \le 2 \cdot 10^5$), the number of test cases. The test cases follow.

The first line of each test case contains two integers: the number of segments n ($1 \le n \le 2 \cdot 10^5$) and the maximum coordinate x ($1 \le x \le 5 \cdot 10^5$).

Next n lines contain the description of segments. Each line contains four integers: the start ℓ_i and end r_i of the segment ($1 \le \ell_i \le r_i \le x$), its weight w_i ($1 \le w_i \le 10^9$) and penalty p_i ($1 \le p_i \le 10^9$).

The sum of n over all test cases does not exceed $2 \cdot 10^5$. The sum of x over all test cases does not exceed $5 \cdot 10^5$.

Output

For each test case, print a line containing one integer: the maximum possible score you can achieve.

Example

standard input	*standard output*
2 3 8 3 7 3 2 5 8 2 1 1 3 2 2 2 5 1 3 2 7 3 5 3 4	16 2

Problem C. Segments and Subsets

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

Consider a collection of segments on a coordinate axis. The coordinates of endpoints are integers from 0 to x. There is no intersecting pair of segments: for any two segments, either one of them contains another, or they have at most one common point.

Your goal is to transform your collection of segments into just a single segment $[0, x]$. No other segments may remain. To achieve this, you can make moves. Each move has one of the following types:

- Select two segments that have a single common point: the right endpoint of the left segment coincides with the left endpoint of the right segment. Merge them into one segment: from leftmost to rightmost point. This move does not cost anything.
- Select one segment. Expand it to the left or to the right by 1 unit. This move costs 1 coin.

If, at some moment of time, there are two or more equal segments, only one of them remains, while the other disappear instantly.

For an initial collection of segments S, let $F(S)$ be the minimum number of coins needed to transform it into just a single segment $[0, x]$. You are given a collection of n segments. Consider all its $2^n - 1$ non-empty sub-collections, calculate F for each of them, and find the sum of these values modulo 998 244 353.

Input

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

Each test case starts with a line containing two integers: the number of segments n ($1 \le n \le 10^5$) and the coordinate x ($1 \le x \le 10^9$). The next n lines describe the segments. The i-th of these lines contains two integers ℓ_i and r_i: the endpoints of the i-th segment ($0 \le \ell_i < r_i \le x$).

The sum of n over all test cases does not exceed 10^5.

Output

For each test case, print a line with a single integer: the required sum modulo 998 244 353.

Example

standard input	*standard output*
3 2 5 1 4 2 3 3 8 1 3 3 5 5 8 7 10 1 5 2 3 3 4 4 5 5 10 6 9 7 8	10 28 806

Problem D. Sum of Characteristics

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

You are given an array a consisting of n random integers from 1 to n. For its subsegment $[\ell, r]$, the *characteristic* is the value

$$C(\ell, r) = \min_{\ell \le i < j \le r} \max(a_i + j, a_j + i).$$

Your task is to calculate

$$\sum_{\ell=1}^{n} \sum_{r=\ell+1}^{n} C(\ell, r).$$

Input

The first line contains an integer t ($1 \le t \le 3 \cdot 10^5$), the number of test cases. The test cases follow.

The first line of each test case contains an integer n, the size of the array ($1 \le n \le 3 \cdot 10^5$). The next line contains the array itself: n integers from 1 to n, picked uniformly and independently by a pseudorandom number generator.

The sum of n over all test cases does not exceed $3 \cdot 10^5$.

Output

For each test case, output a line with a single integer: the sum of characteristics over all the subsegments.

Example

standard input	*standard output*
3 2 2 1 5 3 5 4 1 4 6 1 4 6 1 6 3	4 72 112

Problem E. Random Permutation

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

You are given a permutation p consisting of n integers from 1 to n. You want to build a sequence a from p. To do that, you perform the following operation n times:

- append the minimum element of p to the end of a;
- remove one of the ends of p (either left or right).

You are given a **random** permutation p. Your task is to calculate the number of different sequences a that can be obtained in the way described above. This number can be very large, so find it modulo 998 244 353. Two sequences are different if there is a position at which these sequences differ.

A permutation of size n is a sequence of n distinct integers from 1 to n.

Input

The first line contains an integer t ($1 \le t \le 2 \cdot 10^5$), the number of test cases. The test cases follow.

The first line of each test case contains an integer n, the size of the permutation ($1 \le n \le 2 \cdot 10^5$). The next line contains the permutation itself: n distinct integers from 1 to n. The permutation is generated using a pseudorandom number generator.

The sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a line with a single integer: the required number modulo 998 244 353.

Example

standard input	*standard output*
4 5 4 3 5 1 2 5 5 3 1 2 4 2 2 1 3 1 3 2	8 7 2 4

Problem F. Game

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

Consider an infinite coordinate axis. Flowers bloom in points with coordinates $1, 2, \ldots, n$. The flower at point i has *attractiveness* a_i.

Two players are playing a game. The first player starts at point 1. Then they proceed as follows:

1. The first player, who now stands at point i, picks an integer distance from ℓ_i to r_i, and moves to the right by this distance.
2. If the first player's coordinate is more than n, then the game stops.
3. Otherwise, the second player moves the first player to the left by any distance from 0 to c. However, this move can not end to the left of the point $i + 1$.
4. The first player takes the flower in his current point, and game returns to step 1.

The first player wants to maximize the total attractiveness of the flowers he takes, while the second player wants to minimize it.

Your task is to calculate the final total attractiveness of the flowers the first player has gathered if both players play optimally.

Input

The first line contains an integer t ($1 \le t \le 3 \cdot 10^5$), the number of test cases. The test cases follow.

The first line of each test case contains two integers: the number of flowers n ($1 \le n \le 3 \cdot 10^5$) and the limit c ($0 \le c \le n$). Each of the next three lines contains n integers: these lines describe the arrays ℓ, r, and a, in this order ($1 \le \ell_i \le r_i \le n$; $-10^9 \le a_i \le 10^9$). Note that attractiveness can be negative.

The sum of n over all test cases does not exceed $3 \cdot 10^5$.

Output

For each test case, output a line with a single integer: the final total attractiveness if both players play optimally.

Example

standard input	*standard output*
2 6 2 1 1 2 2 1 1 2 1 3 2 1 1 2 3 1 -1 -1 1 4 4 1 1 1 1 2 2 1 1 -1 -3 -2 -4	3 -9

Problem G. Permutation and Queries

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

You are given a permutation $p_1, p_2, \ldots, p_n$ of size n. Calculate the value

$$f(p) = \min_{i \neq j} |i - j| \cdot |p_i - p_j|.$$

You are also given q queries. The i-th query consists of two indices a_i and b_i. You should swap the elements at these positions (swap p_{a_i} and p_{b_i}), and then recalculate the value $f(p)$. Note that the changes persist between queries: after i-th query, there are i swaps made.

A permutation of size n is a sequence of n distinct integers from 1 to n.

Input

The first line contains two integers: the permutation size n ($2 \le n \le 10^5$) and the number of queries q ($1 \le q \le 10^5$).

The second line describes the permutation p.

Each of the next q lines describes a query. The i-th of these lines contains two integers a_i and b_i ($1 \le a_i, b_i \le n$; $a_i \neq b_i$): the indices of elements you should swap.

Output

Print $q + 1$ lines: the value $f(p)$ before all queries and after each of the q queries.

Example

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

Problem H. Make a Palindrome

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

You have a string s consisting of lowercase English letters. You want to transform it into a palindrome by performing zero or more operations. In one operation, you can swap any two characters in the string which are at distance exactly 2 from each other (in other words, there is exactly one character between them).

Determine if it is possible to transform the string s into a palindrome.

A palindrome is a string that coincides with its reversed copy.

Input

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

The first line of each test case contains an integer n ($1 \le n \le 10^5$). The second line contains the string s of length n consisting of lowercase English letters.

The sum of n over all test cases does not exceed 10^5.

Output

For each test case, print a line containing "`YES`" if it is possible to transform the given string into a palindrome by the given rules, or "`NO`" otherwise.

Example

standard input	*standard output*
8 6 acbbca 6 acbbac 6 aaaaaa 7 abcacba 9 abcbcecea 1 b 2 ca 2 cc	YES NO YES YES YES YES NO YES

Problem I. Good Subsegments

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

You are given an array $a[1..n]$ consisting of n integers from 1 to n. A *subsegment* $a[\ell..r]$ of the array is its consecutive part from position ℓ to position r, inclusive.

A subsegment $a[\ell..r]$ is *k-good* if the following conditions are satisfied:

- $r - \ell + 1 \geq 2 \cdot k$, so its length is at least $2 \cdot k$;
- $a_\ell = a_{\ell+1} = a_{\ell+2} = ... = a_{\ell+k-1}$, so at least k of its leftmost elements are equal to each other;
- $a_r = a_{r-1} = a_{r-2} = ... = a_{r-k+1}$, so at least k its rightmost elements are equal to each other;
- $a_\ell = a_r$, so its ends are equal.

For each k from 1 to $\lfloor \frac{n}{2} \rfloor$, find the number of k-good subsegments of the given array a.

Input

The first line contains an integer t ($1 \leq t \leq 5 \cdot 10^5$), the number of test cases. The test cases follow.

The first line of each test case contains an integer n ($2 \leq n \leq 5 \cdot 10^5$).

The second line consists of n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq n$).

The sum of n over all test cases does not exceed $5 \cdot 10^5$.

Output

For each test case, print a line with $\lfloor \frac{n}{2} \rfloor$ integers: the number of k-good subsegments for each corresponding k, starting from 1.

Example

standard input	*standard output*
4 10 1 2 2 2 2 2 3 2 2 2 6 1 1 1 2 1 1 9 2 2 1 1 1 2 2 1 1 10 3 2 3 2 4 2 10 10 10 10	28 11 3 0 0 10 2 0 16 3 0 0 10 1 0 0 0

Problem J. Series Sum

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

Given are two integers k and p. Calculate

$$\sum_{n=k}^{\infty} \frac{\binom{n}{k}^p}{2^n},$$

where $\binom{n}{k}$ denotes a binomial coefficient that equals to $\frac{n!}{k! \cdot (n-k)!}$.

It is guaranteed that the result can be represented in the form of $\frac{r}{q}$ where r and q are positive coprime integers and $q \neq 0$. Find $r \cdot q^{-1}$ modulo 998 244 353.

Input

The first line contains an integer t ($1 \leq t \leq 2112$), the number of test cases. The test cases follow.

Each test case is described by a single line containing two integers k and p ($k, p \geq 1$; $p \cdot k \leq 10^6$).

The total sum of $p \cdot k$ over all test cases does not exceed 10^6.

Output

For each test case, print a single line containing the integer $r \cdot q^{-1} \bmod 998\,244\,353$: the result of the calculation.

Example

standard input	*standard output*
3 2 3 1 10 9 6	818 204495126 16726290

Problem K. Maximize the Minimum

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

You have an array a of length n and an array b of length m. You can choose to remove some elements from the arrays. Removing element a_i costs c_i coins, and removing element b_j costs d_j coins. Importantly, there should be **at least one** element left in a and **at least one** left in b.

When you are done removing the elements, you compute the following value:

$$\min_{\substack{1 \le i \le n \\ 1 \le j \le m}} |a_i - b_j|.$$

You want to maximize this value. What is the maximum value you can get if you can spend at most s coins in total?

Input

The first line contains an integer t ($1 \le t \le 2 \cdot 10^5$), the number of test cases. The test cases follow.

The first line of each test case contains integers n ($1 \le n \le 2 \cdot 10^5$), m ($1 \le m \le 2 \cdot 10^5$) and s ($0 \le s \le 10^{18}$). The next four lines contain integer arrays a, b, c, d, in this order ($-10^9 \le a_i, b_j \le 10^9$; $1 \le c_i, d_j \le 10^{12}$). The arrays a and c have length n. The arrays b and d have length m.

The sum of n over all test cases does not exceed $2 \cdot 10^5$. The sum of m over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print the maximum possible value you can get.

Example

standard input	*standard output*
2 1 4 10 15 1 6 9 13 8 3 1 2 4 5 4 4 -1 5 3 2 -4 -7 8 6 2 2 3 1 1 2 3 1 1 1	14 3

Problem L. Permutations and Cycles (Maximum Version)

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

For a given x, a permutation of size n is called *good* if for each $1 \le i < n$ the condition $p_i + p_{i+1} \le x$ holds. Find any good permutation with the **maximum** number of cycles.

A permutation of size n is a sequence of n distinct integers from 1 to n.

A cycle of a permutation p is a sequence of indices $i_1, i_2, \ldots, i_k$ such that $p_{i_1} = i_2$, $p_{i_2} = i_3$, ..., $p_{i_k} = i_1$. The cycles obtained by a cyclic shifting of the sequence are considered to be the same.

Input

The first line contains an integer t ($1 \le t \le 2 \cdot 10^5$), the number of test cases. The test cases follow.

Each test case is given on a line with two integers n ($2 \le n \le 2 \cdot 10^5$) and x ($n+1 \le x \le 2 \cdot n - 1$). These constraints guarantee that at least one *good* permutation exists.

The sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print two lines. The first one should contain the maximum number of cycles in a good permutation of length n. The second line should consist of n integers: the permutation itself. If multiple such permutations exist, print any one of them.

Example

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