Codeforces Round #317 [AimFund Thanks-Round] (Div. 2)

A. Arrays

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given two arrays A and B consisting of integers, **sorted in non-decreasing order**. Check whether it is possible to choose k numbers in array A and choose m numbers in array B so that any number chosen in the first array is strictly less than any number chosen in the second array.

Input

The first line contains two integers n_A, n_B ($1 \le n_A, n_B \le 10^5$), separated by a space — the sizes of arrays A and B, correspondingly.

The second line contains two integers k and m ($1 \le k \le n_A, 1 \le m \le n_B$), separated by a space.

The third line contains n_A numbers $a_1, a_2, \ldots a_{n_A}$ ($-10^9 \le a_1 \le a_2 \le \ldots \le a_{n_A} \le 10^9$), separated by spaces — elements of array A.

The fourth line contains n_B integers $b_1, b_2, \ldots b_{n_B}$ ($-10^9 \le b_1 \le b_2 \le \ldots \le b_{n_B} \le 10^9$), separated by spaces — elements of array B.

Output

Print "YES" (without the quotes), if you can choose k numbers in array A and m numbers in array B so that any number chosen in array A was strictly less than any number chosen in array B. Otherwise, print "NO" (without the quotes).

Examples

input
3 3 2 1 1 2 3 3 4 5
output
YES

input
3 3 3 3 1 2 3 3 4 5
output
NO

input
5 2 3 1 1 1 1 1 1 2 2
output
YES

Note

In the first sample test you can, for example, choose numbers 1 and 2 from array A and number 3 from array B (1 < 3 and 2 < 3).

In the second sample test the only way to choose k elements in the first array and m elements in the second one is to choose all numbers in both arrays, but then not all the numbers chosen in A will be less than all the numbers chosen in B: $3 \not< 3$.

B. Order Book

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

In this task you need to process a set of stock exchange orders and use them to create *order book*.

An *order* is an instruction of some participant to buy or sell stocks on stock exchange. The order number i has price p_i, direction d_i — buy or sell, and integer q_i. This means that the participant is ready to buy or sell q_i stocks at price p_i for one stock. A value q_i is also known as a *volume* of an order.

All orders with the same price p and direction d are merged into one *aggregated* order with price p and direction d. The volume of such order is a sum of volumes of the initial orders.

An order book is a list of aggregated orders, the first part of which contains sell orders sorted by price in descending order, the second contains buy orders also sorted by price in descending order.

An order book of depth s contains s best aggregated orders for each direction. A buy order is better if it has higher price and a sell order is better if it has lower price. If there are less than s aggregated orders for some direction then all of them will be in the final order book.

You are given n stock exhange orders. Your task is to print order book of depth s for these orders.

Input

The input starts with two positive integers n and s ($1 \le n \le 1000, 1 \le s \le 50$), the number of orders and the book depth.

Next n lines contains a letter d_i (either 'B' or 'S'), an integer p_i ($0 \le p_i \le 10^5$) and an integer q_i ($1 \le q_i \le 10^4$) — direction, price and volume respectively. The letter 'B' means buy, 'S' means sell. The price of any sell order is higher than the price of any buy order.

Output

Print no more than $2s$ lines with aggregated orders from order book of depth s. The output format for orders should be the same as in input.

Examples

input
6 2 B 10 3 S 50 2 S 40 1 S 50 6 B 20 4 B 25 10
output
S 50 8 S 40 1 B 25 10 B 20 4

Note

Denote *(x, y)* an order with price x and volume y. There are 3 aggregated buy orders (10, 3), (20, 4), (25, 10) and two sell orders (50, 8), (40, 1) in the sample.

You need to print no more than two best orders for each direction, so you shouldn't print the order (10 3) having the worst price among buy orders.

C. Lengthening Sticks

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given three sticks with positive integer lengths of a, b, and c centimeters. You can increase length of some of them by some positive integer number of centimeters (different sticks can be increased by a different length), but in total by at most l centimeters. In particular, it is allowed not to increase the length of any stick.

Determine the number of ways to increase the lengths of some sticks so that you can form from them a non-degenerate (that is, having a positive area) triangle. Two ways are considered different, if the length of some stick is increased by different number of centimeters in them.

Input

The single line contains 4 integers a, b, c, l ($1 \le a, b, c \le 3 \cdot 10^5$, $0 \le l \le 3 \cdot 10^5$).

Output

Print a single integer — the number of ways to increase the sizes of the sticks by the total of at most l centimeters, so that you can make a non-degenerate triangle from it.

Examples

input
1 1 1 2
output
4

input
1 2 3 1
output
2

input
10 2 1 7
output
0

Note

In the first sample test you can either not increase any stick or increase any two sticks by 1 centimeter.

In the second sample test you can increase either the first or the second stick by one centimeter. Note that the triangle made from the initial sticks is degenerate and thus, doesn't meet the conditions.

D. Minimization

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You've got array A, consisting of n integers and a positive integer k. Array A is indexed by integers from 1 to n.

You need to permute the array elements so that value

$$\sum_{i=1}^{n} |A[i] - A[i] + k|$$

became minimal possible. In particular, it is allowed not to change order of elements at all.

Input

The first line contains two integers n, k ($2 \le n \le 3 \cdot 10^5$, $1 \le k \le min(5000, n - 1)$).

The second line contains n integers $A[1], A[2], ..., A[n]$ ($- 10^9 \le A[i] \le 10^9$), separate by spaces — elements of the array A.

Output

Print the minimum possible value of the sum described in the statement.

Examples

input
3 2 1 2 4
output
1

input
5 2 3 -5 3 -5 3
output
0

input
6 3 4 3 4 3 2 5
output
3

Note

In the first test one of the optimal permutations is $1\ 4\ 2$.

In the second test the initial order is optimal.

In the third test one of the optimal permutations is $2\ 3\ 4\ 4\ 3\ 5$.

E. CNF 2

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

'In Boolean logic, a formula is in conjunctive normal form (CNF) or clausal normal form if it is a conjunction of clauses, where a clause is a disjunction of literals' (cited from https://en.wikipedia.org/wiki/Conjunctive_normal_form)

In the other words, CNF is a formula of type $(v_{11} | v_{12} | \ldots | v_{1k_1}) \& (v_{21} | v_{22} | \ldots | v_{2k_2}) \& \ldots \& (v_{l1} | v_{l2} | \ldots | v_{lk_l})$, where & represents a logical "AND" (conjunction), | represents a logical "OR" (disjunction), and v_{ij} are some boolean variables or their negations. Each statement in brackets is called a *clause*, and v_{ij} are called *literals*.

You are given a CNF containing variables $x_1, \ldots, x_m$ and their negations. We know that each variable occurs in at most two clauses (with negation and without negation in total). Your task is to determine whether this CNF is *satisfiable*, that is, whether there are such values of variables where the CNF value is true. If CNF is satisfiable, then you also need to determine the values of the variables at which the CNF is true.

It is guaranteed that each variable occurs at most once in each clause.

Input

The first line contains integers n and m ($1 \le n, m \le 2 \cdot 10^5$) — the number of clauses and the number variables, correspondingly.

Next n lines contain the descriptions of each clause. The i-th line first contains first number k_i ($\boldsymbol{k_i \ge 1}$) — the number of literals in the i-th clauses. Then follow space-separated literals v_{ij} ($1 \le |v_{ij}| \le m$). A literal that corresponds to v_{ij} is $x_{|v_{ij}|}$ either with negation, if v_{ij} is negative, or without negation otherwise.

Output

If CNF is not satisfiable, print a single line "NO" (without the quotes), otherwise print two strings: string "YES" (without the quotes), and then a string of m numbers zero or one — the values of variables in satisfying assignment in the order from x_1 to x_m.

Examples

input
2 2 2 1 -2 2 2 -1
output
YES 11

input
4 3 1 1 1 2 3 -1 -2 3 1 -3
output
NO

input
5 6 2 1 2 3 1 -2 3 4 -3 5 4 6 2 -6 -4 1 5
output
YES 100010

Note

In the first sample test formula is $(x_1 | \neg x_2) \& (x_2 | \neg x_1)$. One of possible answer is $x_1 = TRUE, x_2 = TRUE$.