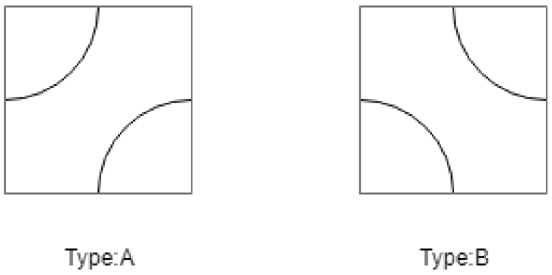
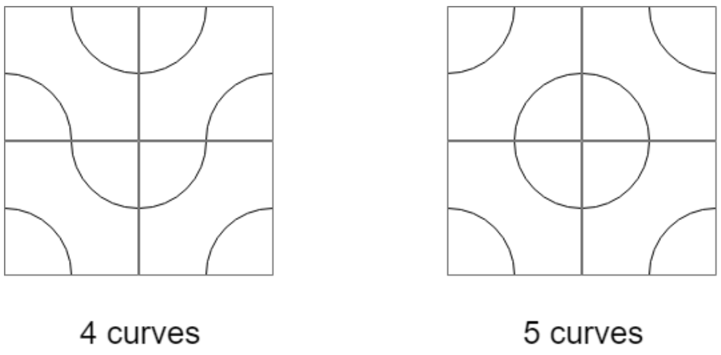
Problem A. Floor Tiles

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Chino has two distinct types of floor tiles, A and B, as depicted in the figure below:



Chino's residence is designed as an $N \cdot M$ planar grid, and she intends to pave it entirely with these two types of tiles. Chino is particularly fascinated by the patterns on the tiles. When the patterns on adjacent tiles align perfectly, they form a continuous curve.



For instance, the left image in the figure above illustrates a configuration with 4 continuous curves, while the right image shows one with 5.

After having already placed the first tile, Chino is curious whether it's possible to continue the tiling in such a way that there are exactly K continuous curves. If such an arrangement exists, your task is to help her find at least one valid solution.

Input

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

Each test case is described as follows:

The first line provides three positive integers $N, M, K(1 \le N, M \le 800, 1 \le K \le 2 \cdot N \cdot M)$, — the dimensions of Chino's house and the desired number of continuous curves.

Subsequently, input two positive integers and a character $x, y, t(1 \le x \le N, 1 \le y \le M, t \in \{'A', 'B'\})$, specifying the coordinates and type of the initially placed tile.

It is guaranteed that the sum of $N \cdot M$ over all test case does not exceed 10^6.

Output

For each test case:

If a solution exists, begin by outputting the word "Yes" to confirm its feasibility.

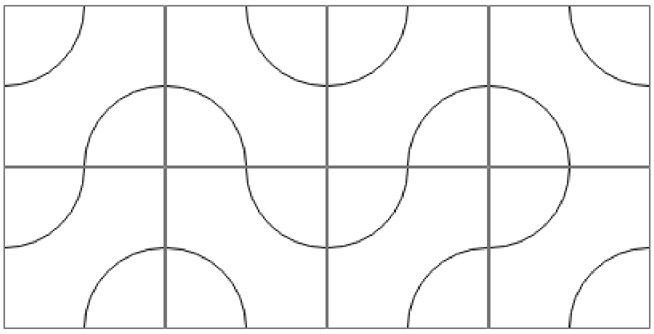
Then, present an $N \cdot M$ characters A or B, which represents one such solution.

Otherwise, output a single line containing the word "No".

Example

standard input	standard output
2 2 4 6 1 1 A 2 4 15 1 1 A	Yes ABAB ABAA No

Note



As shown in the figure

Problem B. MST

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Sajin has recently delved into the study of minimum spanning trees and now he has mastered the algorithm of MST.

He is eager to assess your grasp of minimum spanning tree concepts through a series of queries.

You are confronted with an weighted undirected graph that encompasses n vertices and m edges without any self-loops.

Sajin presents q inquiries. For each, a vertex set S is given. Your objective is to determine the **induced subgraph** of S and find the weight of its minimum spanning tree.

A minimum spanning tree (MST) is a subset of the edges of a connected, edge-weighted graph that connects all the vertices together without any cycles and with the minimum possible total edge weight.

In the mathematical field of graph theory, an **induced subgraph** of a graph is another graph, formed from a subset of the vertices of the graph and all of the edges, from the original graph, connecting pairs of vertices in that subset.

If the **induced subgraph** of S is disconnected, output -1.

Input

The first line contains 3 integers n, m, q $(2 \le n \le 10^5, 1 \le m, q \le 10^5)$, — the number of points, the number of edges, and the number of queries.

Then m lines follow, each line contains three integers u_i, v_i, $w_i(1 \le u_i, v_i \le n,\ u_i \ne v_i,\ 1 \le w_i \le 10^9)$ — the two endpoints of the i-th edge and the edge weight.

Next q lines, each line first contains an integer $k_i(1 \le k_i \le n)$ representing the size of the vertex set S for the i-th query, followed by k_i **distinct** integers $s_{i,j}(1 \le s_{i,j} \le n)$ — the numbers of the vertex set S for the i-th query.

It is guaranteed that the sum of k_i over all queries does not exceed 10^5.

Output

For each query, output one integer representing the answer.

Examples

standard input	standard output

5 10 5
1 2 1
1 3 1
1 4 1
1 5 1
2 3 2
2 4 2
2 5 2
3 4 3
3 5 3
4 5 4
5 1 2 3 4 5
4 2 3 4 5
3 3 4 5
2 4 5
1 5
	4
	6
	6
	4
	0

3 2 1
1 2 1
2 3 1
2 1 3
	-1

Problem C. Red Walking on Grid

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Red is on a $2 \cdot n$ grid, with some cells being red and others being white.

Red can initially choose a red cell, and at each step, can choose a red cell above, below, to the left, or to the right. When Red leaves a cell, the cell immediately turns white.

Red wants to know the maximum number of steps she can take.

If there are no initial red cells, please output 0.

Input

The first line contains a positive integer $n (1 \leq n \leq 10^6)$.

The next two lines contain a $2 \cdot n$ character matrix, consisting only of 'R' and 'W' characters. 'R' represents a red cell, and 'W' represents a white cell.

Output

An integer representing the maximum number of steps Red can take.

Examples

standard input	standard output

`4`
`RWRR`
`RRRR`
	`6`

`4`
`WWWW`
`WWWW`
	`0`

Problem D. Taking Candies

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Emofunc and Cnufome have n boxes of candy, with the i-th box containing a_i candies. They plan to divide these candies.

They will conduct n auctions, and each auction proceeds as follows: Assume Emofunc and Cnufome currently have p and q coins, respectively. They will take turns bidding until one of them gives up.

Specifically, Emofunc starts by placing a bid v_0 $(0 \le v_0 \le p, v_0 \in \mathbb{Z})$.

- For the i-th round $(i = 1, 3, 5, \ldots)$, suppose Emofunc's last bid was v_{i-1}. If $v_{i-1} < q$, Cnufome can bid v_i $(v_{i-1} < v_i \le q, v_i \in \mathbb{Z})$ or choose to give up; if $v_{i-1} \ge q$, Cnufome can only choose to give up.
- For the i-th round $(i = 2, 4, 6, \ldots)$, suppose Cnufome's last bid was v_{i-1}. If $v_{i-1} < p$, Emofunc can bid v_i $(v_{i-1} < v_i \le p, v_i \in \mathbb{Z})$ or choose to give up; if $v_{i-1} \ge p$, Emofunc can only choose to give up.

When anyone gives up, the auction ends. Suppose the final bid was v_m; the last bidder gives v_m coins to the other person and can choose any box of candies from the remaining ones to take.

Initially, Emofunc and Cnufome have x and y coins, respectively. They both know all the information n, x, y, a_i $(1 \le i \le n)$ in advance. Assuming their goal is to obtain as many candies as possible, and they both adopt optimal strategies, how many candies can Emofunc eventually take?

Note that they **don't** care how many coins they end up with, they just want to maximize the number of candies they can get.

Input

The first line contains three integers n, x, y $(1 \le n \le 10^5, 0 \le x, y \le 100)$ — the number of boxes of candy and the number of coins Emofunc and Cnufome have, respectively. The second line contains n integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le 10^5)$ — the number of candies in each box.

Output

An integer indicating the maximum number of candies Emofunc can take.

Examples

standard input	standard output

`3 5 3`
`5 1 2`
	`6`

`3 0 1`
`1 2 3`
	`3`

`10 5 8`
`9 7 3 6 8 8 2 4 4 1`
	`27`

Note

An explanation of the first example:

The following is a possible process.

- In auction 1, Emofunc can place a bid 3, leaving Cnufome no choice but giving up. Emofunc can take the 1st box that has 5 candies. After auction 1, Emofunc has 2 coins and 5 candies, and Cnufome has 6 coins and 0 candy.

- In auction 2, Emofunc can place a bid 2. Then, Cnufome can place a bid 3. Emofunc can only give up, and Cnufome can take the 3rd box that has 2 candies. After auction 2, Emofunc has 5 coins and 5 candies, and Cnufome has 3 coins and 2 candies.
- In auction 3, Emofunc can place a bid 3. Cnufome can only give up, and Emofunc takes the 2nd box that has 1 candy. After auction 3, Emofunc has 2 coins and 6 candies, and Cnufome has 6 coins and 2 candies.

It can be proven that 6 is the maximum number of candies Emofunc can take when both use optimal strategies.

Problem E. GCD VS XOR

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Ben_H has a positive integer x. He wants you to find another positive integer y, which is strictly less than x, so that the equation $gcd(x, y) = x \oplus y$ holds. Can you help him?

where $\oplus$ is bitwise XOR operation (see notes for explanation).

Input

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

The only line of each test case contains a single integer $x(1 \le x \le 10^{18})$.

Output

For each testcase, output a single positive integer y, if you find a feasible answer, or -1 otherwise.

Example

standard input	standard output
2 15 1	10 -1

Note

A bitwise XOR is a binary operation that takes two bit patterns of equal length and performs the logical exclusive OR operation on each pair of corresponding bits. The result in each position is 1 if only one of the bits is 1, but will be 0 if both are 0 or both are 1.

Problem F. Mixed Juice

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

Chino enjoys experimenting with unique juice blends.

There are N types of pre-made juices, numbered from 1 to N.

Chino selects several pre-made juices and mixes them according to specific proportions to create a blend.

Each pre-made specifies its volume w_i and sugar content v_i. Each time Chino selects them, she can freely adjust the values of w_i and v_i any number of times, but she needs to ensure the limitations that $w_i \in [l_i, r_i]$ and $v_i \in [L_i, R_i]$.

More precisely, when she chooses some pre-made juices, first she freely adjusts the values of w_i and v_i within their limitations. Let's say the i-th juice has a volume w_i and sugar content v_i after the adjustment, she can also independently determine proportion coefficients k_i (where k_i is a real number within the range $[0, 1]$) for each juice. The resulting blend will have a volume $W = \sum_{i=1} w_i \cdot k_i$ and a sugar content $V = \sum_{i=1} v_i \cdot k_i$.

Sometimes Chino will change the limitations of pre-made juices.

Occasionally, Chino wants to verify if she can utilize only the pre-made juices numbered from s to t to create a blend with a volume $W = W_i$ and sugar content $V = V_i$.

Input

The first line contains two positive integers N and Q ($1 \le N, Q \le 2 \cdot 10^5$) denoting the count of recipes and operations, respectively.

The subsequent N lines each consist of four positive integers

l_i, r_i, L_i, R_i ($1 \le l_i \le r_i \le 10^9, 1 \le L_i \le R_i \le 10^9$) — the limitations for each pre-made juice.

Following Q lines describe operations op($op \in \{0, 1\}$):

- For $op = 1$, it's followed by five integers

  x, l_i, r_i, L_i, R_i ($1 \le x \le N, 1 \le l_i \le r_i \le 10^9, 1 \le L_i \le R_i \le 10^9$) — an update of the limitations of the x-th pre-made juice to $w_i \in [l_i, r_i]$ and $v_i \in [L_i, R_i]$.
- For $op = 2$, it's followed by four positive integers s_i, t_i, W_i, V_i ($1 \le s_i \le t_i \le N, 1 \le W_i, V_i \le 10^{18}$) — a query to determine if it's feasible to use only pre-made juices numbered from s to t to produce a blend with volume $W = W_i$ and sugar content $V = V_i$.

Output

For each query, output `"Yes"` if creating the blend is feasible; otherwise, output `"No"`.

Example

standard input	standard output

3 6
2 2 3 3
1 3 1 3
1 1 1 1
1 3 4 5 6 7
2 1 1 1 2
2 1 1 2 3
2 1 3 5 9
2 1 3 7 10
2 1 3 10 11
	No
	Yes
	Yes
	Yes
	Yes

Problem G. The Set of Squares

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Define a non-empty positive integer multi-set as "Good" iff the product of all elements in the set is the square of some positive integer x, and the "Good" set has a weight of x.

Let the set SS be defined as the multi-set of all non-empty subsets of the multi-set S.

For example, when $S = \{1_1, 1_2, 8\}$, $SS = \{\{1_1\}, \{1_2\}, \{8\}, \{1_1, 1_2\}, \{1_1, 8\}, \{1_2, 8\}, \{1_1, 1_2, 8\}\}$

Now, given a multi-set S of size N, Chino wants to know the sum of the weights of all "Good" sets in SS. Output the answer modulo $10^9 + 7$.

Input

The first line contains a positive integer $N (1 \le N \le 1000)$ — the size of S.

The next line contains N positive integers $S_i (1 \le S_i \le 1000)$ — each element in the set S.

Output

Only one line with a non-negative integer, representing the answer modulo $10^9 + 7$.

Example

standard input	standard output
6 1 2 6 3 7 7	111

Note

"Good" sets include $\{1\}, \{2, 3, 6\}, \{7_1, 7_2\}, \{1, 2, 3, 6\}, \{1, 7_1, 7_2\}, \{2, 3, 6, 7_1, 7_2\}$

$\{1, 2, 3, 6, 7_1, 7_2\}, \sqrt{1^2} + \sqrt{6^2} + \sqrt{7^2} + \sqrt{6^2} + \sqrt{7^2} + \sqrt{42^2} + \sqrt{42^2} = 111$

Problem H. Instructions Substring

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Red stands at the coordinate $(0, 0)$ of the Cartesian coordinate system. She has a string of instructions: up, down, left, right (where 'right' increases the x-coordinate by 1, and 'up' increases the y-coordinate by 1).

Now Red wants to select a continuous substring of instructions and execute them. Red hopes that the final execution of the instructions can pass through the coordinate (x, y). She wants to know how many selection options there are.

Input

The first line contains three integers n, x, and y ($1 \le n \le 2 \times 10^5, -10^5 \le x, y \le 10^5$) — the length of the instruction string and the coordinates Red hopes to pass through.

The second line contains a string of length n, consisting of the characters $'W'$, $'S'$, $'A'$, and $'D'$ — the four directions: up, down, left, and right, respectively.

Output

Output one integer representing the number of selection options for the continuous substring.

Examples

standard input	standard output

`6 1 1`
`ASAWDD`
	`3`

`4 0 0`
`WWWS`
	`10`

Note

For the first example, choosing $"AWDD"$, $"WD"$, or $"WDD"$ are all valid.

For the second example, any substring can pass through $(0, 0)$, as it is the starting point.

Problem I. Red Playing Cards

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

There are $2 \cdot n$ cards arranged in a row, with each card numbered from 1 to n having exactly 2 copies.

Each time, Red can choose a subarray of consecutive cards (at least 2 cards) to remove from the deck. The chosen subarray must satisfy that the first and last cards have the same number. The score for this operation is: the number of cards multiplied by the number on the first card. Now Red wants to know what is the maximum value of the final score?

Input

The first line contains a positive integer $n(1 \le n \le 3 \times 10^3)$ — the number of card types.

The second line contains $2 \cdot n$ positive integers $a_i(1 \le a_i \le n)$ — the number on each card from left to right.

Output

A single positive integer, representing the maximum final score.

Example

standard input	standard output
4 4 4 1 2 3 1 3 2	21

Note

First operation: choose the 5-th to 7-th cards, get 9 points, and the remaining cards are $[4, 4, 1, 2, 2]$.

Second operation: from the remaining cards, choose the 1-st to 2-nd cards, get 8 points, and the remaining cards are $[1, 2, 2]$.

Third operation: from the remaining cards, choose the 2-nd to 3-rd cards, get 4 points, and there is still 1 card remaining, so no more operations can be performed.

Problem J. Involutions

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

For a set A, a mapping $f : A \to A$ on set A is an involution if and only if for any $x \in A$, $f(f(x)) = x$.

The number of fixed points of f is $FP(f) = |\{x \in A \mid f(x) = x\}|$, where $|\cdot|$ denotes the size of a set. The weight of f is $w(f) = a^{FP(f)}$, where a is a given constant.

Given an integer n, find the sum of the weights $w(f)$ of all involutions f on set $\{1, 2, \ldots, n\}$, modulo 998244353.

Input

The first line contains two integers n ($1 \le n \le 5 \times 10^9$) and a ($1 \le a < 998244353$).

Output

Output one integer indicating the sum of the weights $w(f)$ of all involutions f on set $\{1, 2, \ldots, n\}$, modulo 998244353.

Examples

standard input	standard output
`5 2`	`142`
`10 3`	`1112184`
`114514 1919810`	`786967471`