Problem A. Red-blue table

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Aidos and Tima are going to play an interesting game on a table of sizes $N \times M$. Also, they have an unlimited number of stones of two colors: red and blue. They want to fill the entire table in such a way that each cell of the table contains exactly one stone.

Aidos likes rows of the table with the number of red stones strictly greater than the number of blue stones. Let's denote the number of these rows as A.

Tima likes columns of the table with the number of blue stones strictly greater than the number of red stones. Let's denote the number of these columns as B.

As they are given only one table, they decided not to annoy each other and fill the table such that the total number of rows that Aidos likes and columns that Tima likes would be as many as possible.

Formally, they will try to maximize the value of the expression $A + B$.

Help the guys to fill the table.

Input

The first line contains a single integer T $(1 \le T \le 1000)$ — the number of tests.

Next T lines contain two integers numbers N, M $(1 \le N, M \le 1000)$. It is guaranteed that the sum of $N \cdot M$ over all the tests will not exceed 10^6.

Output

The answer for each test consists of $N + 1$ lines. Output the maximum value of $A + B$ in one line. In each of the next N lines output M symbols ('+' — for a red stone, '-' — for a blue stone). If there are several solutions you can output any of them.

Scoring

This task contains six substasks:

1. $1 \le T \le 16, 1 \le N, M \le 4$. Scored 17 points.
2. $1 \le T \le 1000, 1 \le N, M \le 50, min(N, M) \le 3$. Scored 10 points.
3. $1 \le T \le 1000, 1 \le N, M \le 50, min(N, M) \le 5$. Scored 16 points.
4. $1 \le T \le 1000, 1 \le N, M \le 1000$. N and M — odd numbers. Scored 11 points.
5. $1 \le T \le 1000, 1 \le N, M \le 1000, N = M$. Scored 15 points.
6. $1 \le T \le 1000, 1 \le N, M \le 1000$. Scored 31 points.

Example

<table>
<tr><th>standard input</th><th>standard output</th></tr>
<tr><td><pre>2
1 3
3 3</pre></td><td><pre>3

4
+-+
+-+
+++</pre></td></tr>
</table>

Problem B. Hedgehog Daniyar and Algorithms

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

Hedgehog Daniyar wants to learn new algorithms. To help his friend, Invisible Zhanadil gave Daniyar N algorithmic books, each book having its own weight w_i $(1 \le i \le N)$. Hedgehog Daniyar arranged the books from 1 to N on the shelf.

Hedgehog Daniyar's learning journey is spread out to M days: during day i, he is interested in reading the books from l_i to r_i. As a perfectionist, he first tries to rearrange the books from l to r in non-decreasing order of their weights. To achieve that, the hedgehog can swap **any two adjacent books** within the range l_i and r_i as long as **their total weight doesn't exceed** his mood k_i. Luckily, he already knows his mood for each of the upcoming M days. At the end of each day, again due to his perfectionism, he returns all the books back to their original positions.

Help the hedgehog to improve his plan - find out for each day whether his mood is enough to rearrange books in non-decreasing order of their weights.

For example, assume that hedgehog Daniyar is planning to read three books, currently arranged as $[3, 5, 4]$ and his mood is 8. Then, sadly, it's not possible since he can't swap books with weights 5 and 4 (because $5 + 4 > 8$). But if his mood is 9, then it's possible to rearrange the books in non-decreasing order of their weights.

Note that each day is independent of other days, meaning that at the start of each day arrangement of books will be **in its original state**.

Input

The first line of input contains two integers N, M $(1 \le N, M \le 10^6)$ — the number of algorithmic books and the number of days.

The second line of input contains N integers $w_1, w_2, ..., w_N (0 \le w_i \le 10^9$ for all $1 \le i \le N)$ separated with a single space — weight of each book.

Next M lines contain three integers l_i, r_i, and k_i $(1 \le l_i \le r_i \le N$ and $0 \le k_i \le 2 \cdot 10^9)$. Hedgehog Daniyar plans to read the books from l_i to r_i with mood k_i on specific day i.

Output

Output M lines, each containing a single digit. The line i should contain 1 if it's possible for hedgehog Daniyar to read those books on day i and 0 otherwise.

Scoring

This task contains six sub-tasks:

1. $1 \le N, M \le 500$. Scored 8 points.
2. $1 \le N, M \le 5000$. Scored 9 points.
3. $1 \le N, M \le 10^6, 0 \le k < w_i$ where $1 \le i \le N$. Scored 13 points.
4. $1 \le N, M \le 10^5, 0 \le w_i \le 1000$. Scored 17 points.
5. $1 \le N, M \le 2 \cdot 10^5$. Scored 30 points.
6. Constraints from problem statement above. Scored 23 points.

Example

standard input	standard output
5 2 3 5 1 8 2 1 3 6 2 5 3	1 0

Note

In the first query, Hedgehog Daniyar can achieve the right arrangement in the following way:

$[3, \mathbf{5}, \mathbf{1}, 8, 2]$

$[\mathbf{3}, \mathbf{1}, 5, 8, 2]$

$[1, 3, 5, 8, 2]$

Problem C. Intergalactic ship

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given a sequence a of n integer numbers $a_1, a_2, ..., a_n$.

In addition, you are given a set S of q updates. Each update is defined by three numbers l, r, and x. An update consists of the operation xor with the number x applied to all the numbers in the segment $[l, r]$ of the sequence a. Formally, for each $l \le i \le r$ the following substitution is performed:

$$a_i := a_i \oplus x$$

For a set of updates S, let's define $K(S)$ as the sum of $sum(i,j)^2$ over all possible segments of the sequence a after applying all updates from the set S to the given sequence:

$$K(S) = \sum_{1 \le i \le j \le n} sum(i,j)^2$$

where $sum(i,j)$ is defined as the sum of elements in the segment $[i,j]$:

$$sum(i,j) = \sum_{x=i}^{j} a_x$$

Your task is to find the sum over all 2^q subsets of the given set of updates S. Formally, if P is the set of all subsets of the set S of q updates, you have to find the following:

$$\sum_{subset \in P} K(subset)$$

Input

The first line of input contains single integer n $(1 \le n \le 1000)$ — the number of elements in the sequence.

The second line contains n space-separated integer numbers $a_1, a_2, ..., a_n$ $(0 \le a_i < 128$ for each $1 \le i \le n)$ — the given sequence.

The third line contains single integer q $(1 \le q \le 10^5)$ — the number of updates.

Each of the next q lines contains three space-separated integer numbers l, r, and x $(1 \le l \le r \le n,\ 0 \le x < 128)$ — descriptions of the updates.

Output

Output single integer — answer to the problem. As soon as the answer may be very large, output it modulo $10^9 + 7$.

Scoring

This problem consists of nine subtasks:

1. $1 \leq n \leq 10$, $1 \leq q \leq 10$, $0 \leq a_i, x < 128$, for all $1 \leq i \leq n$. Scored 4 points.
2. $1 \leq n \leq 100$, $1 \leq q \leq 10$, $0 \leq a_i, x < 128$, for all $1 \leq i \leq n$. Scored 5 points.
3. $1 \leq n \leq 100$, $1 \leq q \leq 100000$, $0 \leq a_i, x < 32$, for all $1 \leq i \leq n$. It is guaranteed that length of all update segments is equal to 1. Scored 6 points.
4. $1 \leq n \leq 1000$, $1 \leq q \leq 500$, $0 \leq a_i, x < 128$, for all $1 \leq i \leq n$. It is guarenteed that all update segments do not intersect pairwise. Scored 9 points.
5. $1 \leq n \leq 30$, $1 \leq q \leq 20$, $0 \leq a_i, x < 32$, for all $1 \leq i \leq n$. Scored 8 points.
6. $1 \leq n \leq 30$, $1 \leq q \leq 5000$, $0 \leq a_i, x < 32$, for all $1 \leq i \leq n$. Scored 11 points.
7. $1 \leq n \leq 300$, $1 \leq q \leq 300$, $0 \leq a_i, x < 128$, for all $1 \leq i \leq n$. Scored 19 points.
8. $1 \leq n \leq 500$, $1 \leq q \leq 100000$, $0 \leq a_i, x < 128$, for all $1 \leq i \leq n$. Scored 30 points.
9. $1 \leq n \leq 1000$, $1 \leq q \leq 100000$, $0 \leq a_i, x < 128$, for all $1 \leq i \leq n$. Scored 8 points.

Examples

standard input	standard output
2 1 3 1 1 2 2	52
5 1 2 3 4 5 0	1001

Note

The xor operation is the bitwise exclusive OR.

In the first sample, there are $2^1 = 2$ possible sequences after applying updates — with applying the single given operation and without. In both sequences the resulting sums are equal to 26.

In the second sample, set S is empty, the set of all subsets consists of a single element $\varnothing$ — empty set, i.e. there are no updates and you have to find $K(\varnothing)$ for the given sequence a.