Sponsored by Telegram

Codeforces Round #775 (Div. 2, based on Moscow Open Olympiad in Informatics)

A. Game

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

You are playing a very popular computer game. The next level consists of n consecutive locations, numbered from 1 to n, each of them containing either land or water. It is known that the first and last locations contain land, and for completing the level you have to move from the first location to the last. Also, if you become inside a location with water, you will die, so you can only move between locations with land.

You can jump between adjacent locations for free, as well as **no more than** once jump from any location with land i to any location with land $i + x$, spending x coins ($x \geq 0$).

Your task is to spend the minimum possible number of coins to move from the first location to the last one.

Note that this is always possible since both the first and last locations are the land locations.

Input

There are several test cases in the input data. The first line contains a single integer t ($1 \leq t \leq 100$) — the number of test cases. This is followed by the test cases description.

The first line of each test case contains one integer n ($2 \leq n \leq 100$) — the number of locations.

The second line of the test case contains a sequence of integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i \leq 1$), where $a_i = 1$ means that the i-th location is the location with land, and $a_i = 0$ means that the i-th location is the location with water. It is guaranteed that $a_1 = 1$ and $a_n = 1$.

Output

For each test case print a single integer — the answer to the problem.

Example

input

```
3
2
1 1
5
1 0 1 0 1
4
1 0 1 1
```

output

```
0
4
2
```

Note

In the first test case, it is enough to make one free jump from the first location to the second one, which is also the last one, so the answer is 0.

In the second test case, the only way to move from the first location to the last one is to jump between them, which will cost 4 coins.

In the third test case, you can jump from the first location to the third for 2 coins, and then jump to the fourth location for free, so the answer is 2. It can be shown that this is the optimal way.

B. Game of Ball Passing

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

Daniel is watching a football team playing a game during their training session. They want to improve their passing skills during that session.

The game involves n players, making multiple passes towards each other. Unfortunately, since the balls were moving too fast, after the session Daniel is unable to know how many balls were involved during the game. The only thing he knows is the number of passes delivered by each player during all the session.

Find the minimum possible amount of balls that were involved in the game.

Input

There are several test cases in the input data. The first line contains a single integer t ($1 \le t \le 5 \cdot 10^4$) — the number of test cases. This is followed by the test cases description.

The first line of each test case contains one integer n ($2 \le n \le 10^5$) — the number of players.

The second line of the test case contains a sequence of integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$), where a_i is the number of passes delivered by the i-th player.

It is guaranteed that the sum of n over all test cases doesn't exceed 10^5.

Output

For each test case print a single integer — the answer to the problem.

Example

input

```
4
4
2 3 3 2
3
1 5 2
2
0 0
4
1000000000 1000000000 1000000000 1000000000
```

output

```
1
2
0
1
```

Note

In the first test case, with the only ball, the game can go like this:

$2 \rightarrow 1 \rightarrow 3 \rightarrow 4 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 2 \rightarrow 3 \rightarrow 2.$

In the second test case, there is no possible way to play the game with only one ball. One possible way to play with two balls:

$2 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 1.$

$2 \rightarrow 3 \rightarrow 2 \rightarrow 1$

In the third example, there were no passes, so 0 balls are possible.

C. Weird Sum

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

Egor has a table of size $n \times m$, with lines numbered from 1 to n and columns numbered from 1 to m. Each cell has a color that can be presented as an integer from 1 to 10^5.

Let us denote the cell that lies in the intersection of the r-th row and the c-th column as (r, c). We define the manhattan distance between two cells (r_1, c_1) and (r_2, c_2) as the length of a shortest path between them where each consecutive cells in the path must have a common side. The path can go through cells of any color. For example, in the table 3×4 the manhattan distance between $(1, 2)$ and $(3, 3)$ is 3, one of the shortest paths is the following: $(1, 2) \rightarrow (2, 2) \rightarrow (2, 3) \rightarrow (3, 3)$.

Egor decided to calculate the sum of manhattan distances between each pair of cells of the same color. Help him to calculate this sum.

Input

The first line contains two integers n and m ($1 \le n \le m$, $n \cdot m \le 100\,000$) — number of rows and columns in the table.

Each of next n lines describes a row of the table. The i-th line contains m integers $c_{i1}, c_{i2}, \ldots, c_{im}$ ($1 \le c_{ij} \le 100\,000$) — colors of cells in the i-th row.

Output

Print one integer — the the sum of manhattan distances between each pair of cells of the same color.

Examples

input

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

output

```
7
```

input

```
3 4
1 1 2 2
2 1 1 2
2 2 1 1
```

output

```
76
```

input

```
4 4
1 1 2 3
2 1 1 2
3 1 2 1
1 1 2 1
```

output

```
129
```

Note

In the first sample there are three pairs of cells of same color: in cells $(1, 1)$ and $(2, 3)$, in cells $(1, 2)$ and $(2, 2)$, in cells $(1, 3)$ and $(2, 1)$. The manhattan distances between them are 3, 1 and 3, the sum is 7.

D. Integral Array

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given an array a of n positive integers numbered from 1 to n. Let's call an array *integral* if for any two, not necessarily different, numbers x and y from this array, $x \ge y$, the number $\left\lfloor \frac{x}{y} \right\rfloor$ (x divided by y with rounding down) is also in this array.

You are guaranteed that all numbers in a do not exceed c. Your task is to check whether this array is integral.

Input

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

The first line of each test case contains two integers n and c ($1 \le n \le 10^6$, $1 \le c \le 10^6$) — the size of a and the limit for the numbers in the array.

The second line of each test case contains n integers a_1, a_2, ..., a_n ($1 \le a_i \le c$) — the array a.

Let N be the sum of n over all test cases and C be the sum of c over all test cases. It is guaranteed that $N \le 10^6$ and $C \le 10^6$.

Output

For each test case print Yes if the array is integral and No otherwise.

Examples

input

```
4
3 5
1 2 5
4 10
1 3 3 7
1 2
2
1 1
1
```

output

```
Yes
No
No
Yes
```

input

```
1
1 1000000
1000000
```

output

```
No
```

Note

In the first test case it is easy to see that the array is integral:

- $\left\lfloor \frac{1}{1} \right\rfloor = 1$, $a_1 = 1$, this number occurs in the arry
- $\left\lfloor \frac{2}{2} \right\rfloor = 1$
- $\left\lfloor \frac{5}{5} \right\rfloor = 1$
- $\left\lfloor \frac{2}{1} \right\rfloor = 2$, $a_2 = 2$, this number occurs in the array
- $\left\lfloor \frac{5}{1} \right\rfloor = 5$, $a_3 = 5$, this number occurs in the array
- $\left\lfloor \frac{5}{2} \right\rfloor = 2$, $a_2 = 2$, this number occurs in the array

Thus, the condition is met and the array is integral.

In the second test case it is enough to see that

$\left\lfloor \frac{7}{3} \right\rfloor = \left\lfloor 2\frac{1}{3} \right\rfloor = 2$, this number is not in a, that's why it is not integral.

In the third test case $\left\lfloor \frac{2}{2} \right\rfloor = 1$, but there is only 2 in the array, that's why it is not integral.

E. Tyler and Strings

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

While looking at the kitchen fridge, the little boy Tyler noticed magnets with symbols, that can be aligned into a string s.

Tyler likes strings, and especially those that are lexicographically smaller than another string, t. After playing with magnets on the fridge, he is wondering, how many distinct strings can be composed out of letters of string s by rearranging them, so that the resulting string is lexicographically smaller than the string t? Tyler is too young, so he can't answer this question. The alphabet Tyler uses is very large, so for your convenience he has already replaced the same letters in s and t to the same integers, keeping that different letters have been replaced to different integers.

We call a string x lexicographically smaller than a string y if one of the followings conditions is fulfilled:

- There exists such position of symbol m that is presented in both strings, so that before m-th symbol the strings are equal, and the m-th symbol of string x is smaller than m-th symbol of string y.
- String x is the prefix of string y and $x \neq y$.

Because the answer can be too large, print it modulo $998\,244\,353$.

Input

The first line contains two integers n and m ($1 \le n, m \le 200\,000$) — the lengths of strings s and t respectively.

The second line contains n integers $s_1, s_2, s_3, \ldots, s_n$ ($1 \le s_i \le 200\,000$) — letters of the string s.

The third line contains m integers $t_1, t_2, t_3, \ldots, t_m$ ($1 \le t_i \le 200\,000$) — letters of the string t.

Output

Print a single number — the number of strings lexicographically smaller than t that can be obtained by rearranging the letters in s, modulo $998\,244\,353$.

Examples

input

```
3 4
1 2 2
2 1 2 1
```

output

```
2
```

input

```
4 4
1 2 3 4
4 3 2 1
```

output

```
23
```

input

```
4 3
1 1 1 2
1 1 2
```

output

```
1
```

Note

In the first example, the strings we are interested in are $[1\,2\,2]$ and $[2\,1\,2]$. The string $[2\,2\,1]$ is lexicographically larger than the string $[2\,1\,2\,1]$, so we don't count it.

In the second example, all strings count except $[4\,3\,2\,1]$, so the answer is $4! - 1 = 23$.

In the third example, only the string $[1\,1\,1\,2]$ counts.

F. Serious Business

time limit per test: 5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Dima is taking part in a show organized by his friend Peter. In this show Dima is required to cross a $3 \times n$ rectangular field. Rows are numbered from 1 to 3 and columns are numbered from 1 to n.

The cell in the intersection of the i-th row and the j-th column of the field contains an integer $a_{i,j}$. Initially Dima's score equals zero, and whenever Dima reaches a cell in the row i and the column j, his score changes by $a_{i,j}$. Note that the score can become negative.

Initially all cells in the first and the third row are marked as available, and all cells in the second row are marked as unavailable. However, Peter offered Dima some help: there are q special offers in the show, the i-th special offer allows Dima to mark cells in the second row between l_i and r_i as available, though Dima's score reduces by k_i whenever he accepts a special offer. Dima is allowed to use as many special offers as he wants, and might mark the same cell as available multiple times.

Dima starts his journey in the cell $(1, 1)$ and would like to reach the cell $(3, n)$. He can move either down to the next row or right to the next column (meaning he could increase the current row or column by 1), thus making $n + 1$ moves in total, out of which exactly $n - 1$ would be horizontal and 2 would be vertical.

Peter promised Dima to pay him based on his final score, so the sum of all numbers of all visited cells minus the cost of all special offers used. Please help Dima to maximize his final score.

Input

The first input line contains two integers n and q ($1 \le n, q \le 500\,000$) — the number of columns in the field and the number of special offers.

The next three lines describe the field, i-th of them contains n integers a_{i1}, a_{i2}, ..., a_{in} ($-10^9 \le a_{ij} \le 10^9$) — the values in the i-th row.

The next q lines describe special offers: the i-th offer is described by 3 integers l_i, r_i and k_i ($1 \le l_i \le r_i \le n$, $1 \le k_i \le 10^9$) — the segment that becomes unblocked and the cost of this special offer.

Output

Output one integer — the maximum final score Dima can achieve.

Examples

input

```
4 3
1 0 2 -1
-3 1 9 2
3 2 4 1
1 2 5
2 3 4
1 4 14
```

output

```
13
```

input

```
5 4
-20 -10 -11 -10 1
1 3 3 6 3
14 -20 3 6 2
1 5 13
1 2 2
3 5 3
2 3 1
```

output

```
-4
```

Note

In the first example, it is optimal to use Peter's second offer of 4 rubles and go through the cells $(1, 1)$, $(1, 2)$, $(1, 3)$, $(2, 3)$, $(3, 3)$, $(3, 4)$, earning $1 + 0 + 2 + 9 + 4 + 1 - 4 = 13$ rubles in total.

In the second example, it is optimal to use Peter's second and third offers of 2 and 3 rubles, respectively, and go through the cells $(1, 1)$, $(2, 1)$, $(2, 2)$, $(2, 3)$, $(2, 4)$, $(3, 4)$, $(3, 5)$, earning $-20 + 1 + 3 + 3 + 6 + 6 + 2 - 2 - 3 = -4$ rubles.