Codeforces Round #810 (Div. 1)

A. Color the Picture

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

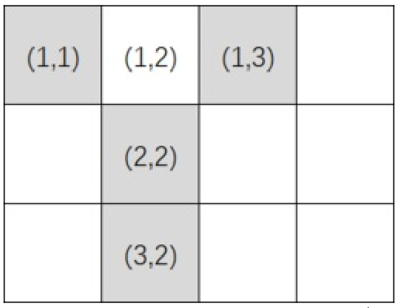
A picture can be represented as an $n \times m$ grid (n rows and m columns) so that each of the $n \cdot m$ cells is colored with one color. You have k pigments of different colors. You have a limited amount of each pigment, more precisely you can color at most a_i cells with the i-th pigment.

A picture is considered beautiful if each cell has **at least** 3 **toroidal** neighbors with the same color as itself.

Two cells are considered **toroidal** neighbors if they **toroidally** share an edge. In other words, for some integers $1 \leq x_1, x_2 \leq n$ and $1 \leq y_1, y_2 \leq m$, the cell in the x_1-th row and y_1-th column is a toroidal neighbor of the cell in the x_2-th row and y_2-th column if one of following two conditions holds:

- $x_1 - x_2 \equiv \pm 1 \pmod{n}$ and $y_1 = y_2$, or
- $y_1 - y_2 \equiv \pm 1 \pmod{m}$ and $x_1 = x_2$.

Notice that each cell has exactly 4 toroidal neighbors. For example, if $n = 3$ and $m = 4$, the toroidal neighbors of the cell $(1, 2)$ (the cell on the first row and second column) are: $(3, 2)$, $(2, 2)$, $(1, 3)$, $(1, 1)$. They are shown in gray on the image below:



The gray cells show toroidal neighbors of $(1, 2)$.

Is it possible to color all cells with the pigments provided and create a beautiful picture?

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^4$). The description of the test cases follows.

The first line of each test case contains three integers n, m, and k ($3 \leq n, m \leq 10^9$, $1 \leq k \leq 10^5$) — the number of rows and columns of the picture and the number of pigments.

The next line contains k integers $a_1, a_2, \dots, a_k$ ($1 \leq a_i \leq 10^9$) — a_i is the maximum number of cells that can be colored with the i-th pigment.

It is guaranteed that the sum of k over all test cases does not exceed 10^5.

Output

For each test case, print "Yes" (without quotes) if it is possible to color a beautiful picture. Otherwise, print "No" (without quotes).

Example

input

```
6
4 6 3
12 9 8
3 3 2
8 8
3 3 2
9 5
4 5 2
10 11
5 4 2
9 11
10 10 3
11 45 14
```

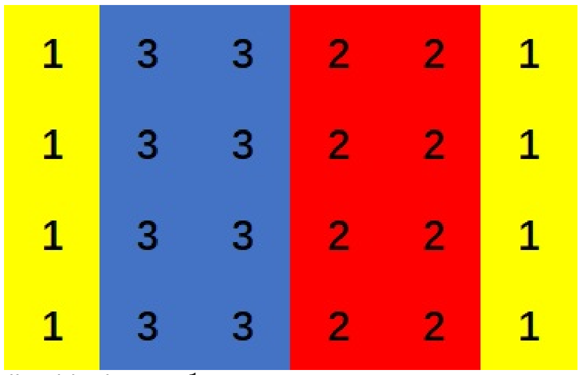
output

```
Yes
No
Yes
```

```
Yes
No
No
```

Note

In the first test case, one possible solution is as follows:



In the third test case, we can color all cells with pigment 1.

B. Rain

time limit per test: 4 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You are the owner of a harvesting field which can be modeled as an infinite line, whose positions are identified by integers.

It will rain for the next n days. On the i-th day, the rain will be centered at position x_i and it will have intensity p_i. Due to these rains, some rainfall will accumulate; let a_j be the amount of rainfall accumulated at integer position j. Initially a_j is 0, and it will increase by $\max(0, p_i - |x_i - j|)$ after the i-th day's rain.

A flood will hit your field if, at any moment, there is a position j with accumulated rainfall $a_j > m$.

You can use a magical spell to erase **exactly one** day's rain, i.e., setting $p_i = 0$. For each i from 1 to n, check whether in case of erasing the i-th day's rain there is no flood.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n and m ($1 \le n \le 2 \cdot 10^5$, $1 \le m \le 10^9$) — the number of rainy days and the maximal accumulated rainfall with no flood occurring.

Then n lines follow. The i-th of these lines contains two integers x_i and p_i ($1 \le x_i, p_i \le 10^9$) — the position and intensity of the i-th day's rain.

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

Output

For each test case, output a binary string s length of n. The i-th character of s is 1 if after erasing the i-th day's rain there is **no** flood, while it is `0`, if after erasing the i-th day's rain the flood still happens.

Example

input

```
4
3 6
1 5
5 5
3 4
2 3
1 3
5 2
2 5
1 6
10 6
6 12
4 5
1 6
12 5
5 5
9 7
8 3
```

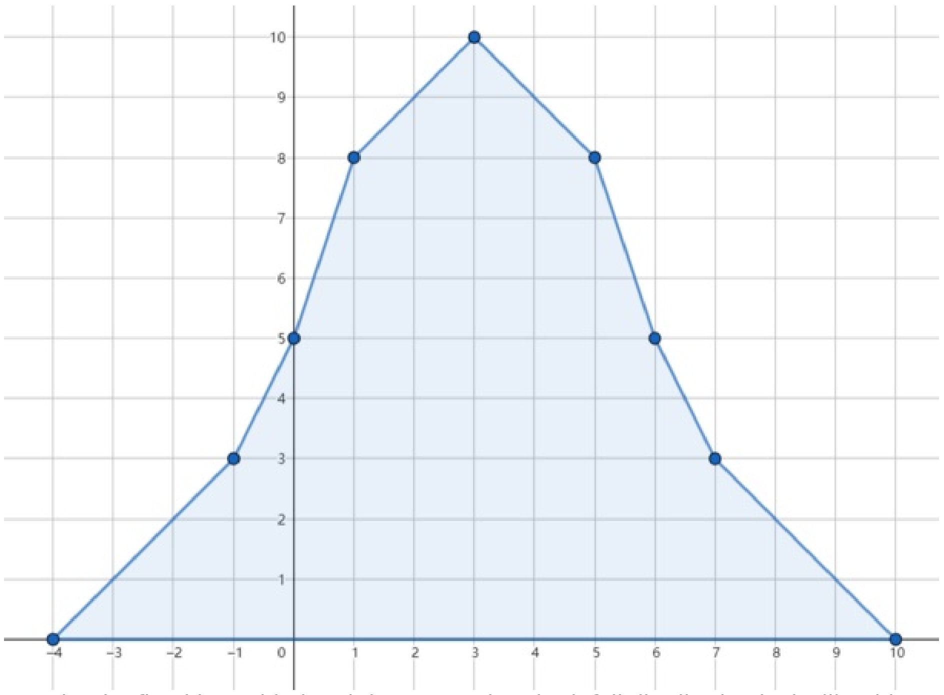
output

```
001
11
```

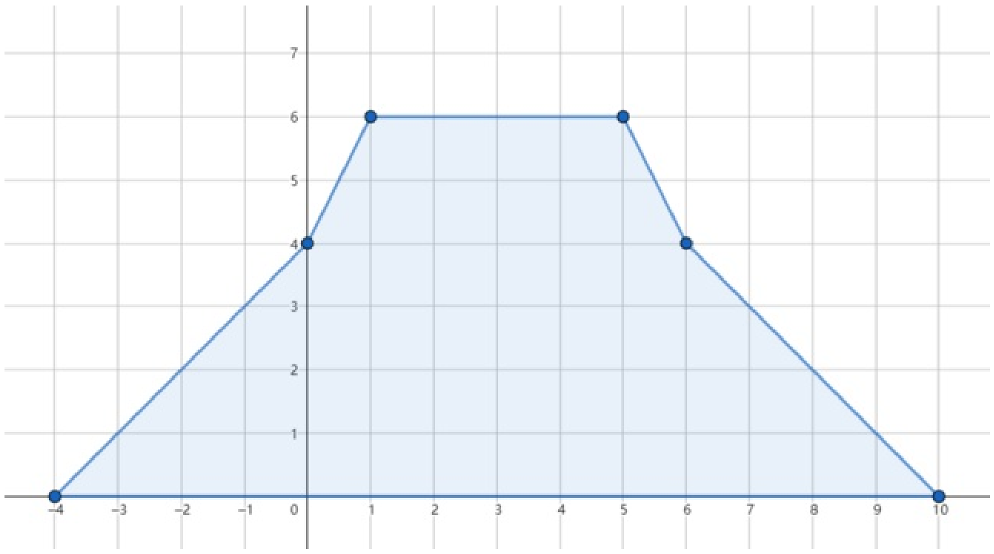
```
00
100110
```

Note

In the first test case, if we do not use the spell, the accumulated rainfall distribution will be like this:



If we erase the third day's rain, the flood is avoided and the accumulated rainfall distribution looks like this:



In the second test case, since initially the flood will not happen, we can erase any day's rain.

In the third test case, there is no way to avoid the flood.

C. XOR Triangle

time limit per test: 4 seconds

memory limit per test: 512 megabytes

input: standard input

output: standard output

You are given a positive integer n. Since n may be very large, you are given its binary representation.

You should compute the number of triples (a, b, c) with $0 \le a, b, c \le n$ such that $a \oplus b$, $b \oplus c$, and $a \oplus c$ are the sides of a non-degenerate triangle.

Here, $\oplus$ denotes the bitwise XOR operation.

You should output the answer modulo $998\,244\,353$.

Three positive values x, y, and z are the sides of a non-degenerate triangle if and only if $x + y > z$, $x + z > y$, and $y + z > x$.

Input

The first and only line contains the binary representation of an integer n $(0 < n < 2^{200\,000})$ without leading zeros.

For example, the string `10` is the binary representation of the number 2, while the string `1010` represents the number 10.

Output

Print one integer — the number of triples (a, b, c) satisfying the conditions described in the statement modulo $998\,244\,353$.

Examples

input
```
101
```
output
```
12
```

input
```
1110
```
output
```
780
```

input
```
11011111101010010
```
output
```
141427753
```

Note

In the first test case, $101_2 = 5$.

- The triple $(a, b, c) = (0, 3, 5)$ is valid because $(a \oplus b, b \oplus c, c \oplus a) = (3, 6, 5)$ are the sides of a non-degenerate triangle.
- The triple $(a, b, c) = (1, 2, 4)$ is valid because $(a \oplus b, b \oplus c, c \oplus a) = (3, 6, 5)$ are the sides of a non-degenerate triangle.

The 6 permutations of each of these two triples are all the valid triples, thus the answer is 12.

In the third test case, $11\,011\,111\,101\,010\,010_2 = 114\,514$. The full answer (before taking the modulo) is $1\,466\,408\,118\,808\,164$.

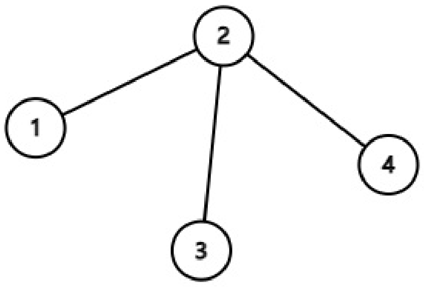
D. Recover the Tree

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Rhodoks has a tree with n vertices, but he doesn't remember its structure. The vertices are indexed from 1 to n.

A segment $[l, r]$ ($1 \le l \le r \le n$) is good if the vertices with indices l, $l + 1$, ..., r form a connected component in Rhodoks' tree. Otherwise, it is bad.

For example, if the tree is the one in the picture, then only the segment $[3, 4]$ is bad while all the other segments are good.



For each of the $\frac{n(n+1)}{2}$ segments, Rhodoks remembers whether it is good or bad. Can you help him recover the tree? If there are multiple solutions, print any.

It is guaranteed that the there is at least one tree satisfying Rhodoks' description.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 1000$). The description of the test cases follows.

The first line of each test case contains an integer n ($1 \le n \le 2000$) — the number of vertices in the tree.

Then n lines follow. The i-th of these lines contains a string $good_i$ of length $n + 1 - i$ consisting of `0` and `1`. If the segment $[i, i + j - 1]$ is good then the j-th character of $good_i$ is `1`, otherwise j-th character of $good_i$ is `0`.

It is guaranteed that the there is at least one tree consistent with the given data.

It is guaranteed that the sum of n over all test cases does not exceed 2000.

Output

For each test case, print $n-1$ lines describing the tree you recover.

The i-th line should contain two integers u_i and v_i ($1 \le u_i, v_i \le n$), denoting an edge between vertices u_i and v_i.

If there are multiple solutions, print any.

Example

input

```
3
4
1111
111
10
1
6
111111
11111
1111
111
11
1
12
100100000001
11100000001
1000000000
100000000
10010001
1110000
100000
10000
1001
111
10
1
```

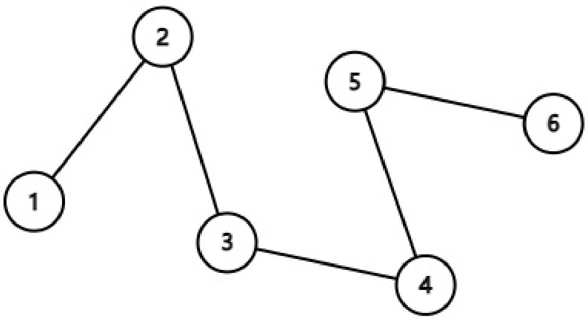
output

```
1 2
2 3
2 4
1 2
2 3
3 4
4 5
5 6
2 3
6 7
10 11
2 4
6 8
10 12
1 4
5 8
9 12
5 12
2 12
```

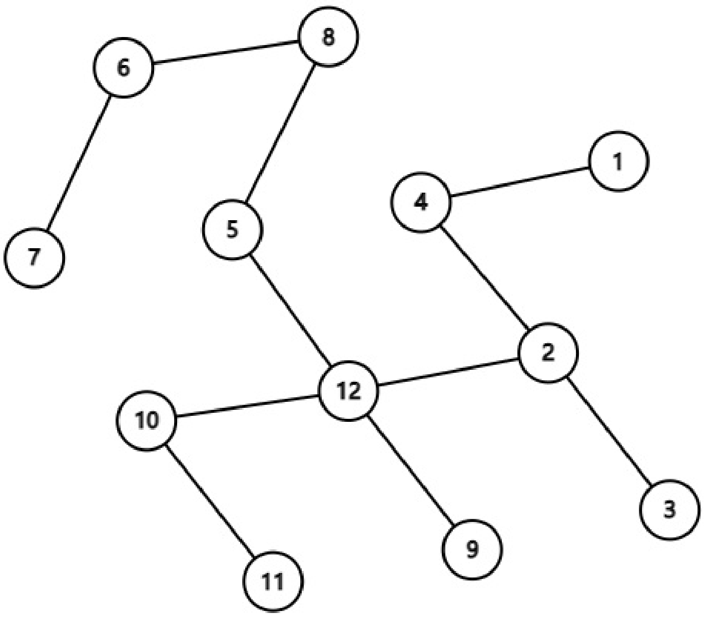
Note

The first test case is explained in the statement.

In the second test case, one possible tree is as follows:



In the third test case, one possible tree is as follows:



E. Two Arrays

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

You are given two arrays of integers $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_m$.

Alice and Bob are going to play a game. Alice moves first and they take turns making a move.

They play on a grid of size $n \times m$ (a grid with n rows and m columns). Initially, there is a rook positioned on the first row and first column of the grid.

During her/his move, a player can do one of the following two operations:

1. Move the rook to a **different** cell on the same row or the same column of the current cell. A player cannot move the rook to a

cell that has been visited 1000 times before (i.e., the rook can stay in a certain cell at most 1000 times during the entire game). Note that the starting cell is considered to be visited once at the beginning of the game.

2. End the game immediately with a score of $a_r + b_c$, where (r, c) is the current cell (i.e., the rook is on the r-th row and c-th column).

Bob wants to maximize the score while Alice wants to minimize it. If they both play this game optimally, what is the final score of the game?

Input

The first line contains two integers n and m $(1 \le n, m \le 2 \cdot 10^5)$ — the length of the arrays a and b (which coincide with the number of rows and columns of the grid).

The second line contains the n integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 5 \cdot 10^8)$.

The third line contains the m integers $b_1, b_2, \ldots, b_n$ $(1 \le b_i \le 5 \cdot 10^8)$.

Output

Print a single line containing the final score of the game.

Examples

input

```
2 1
3 2
2
```

output

```
4
```

input

```
4 5
235499701 451218171 355604420 132973458
365049318 264083156 491406845 62875547 175951751
```

output

```
531556171
```

Note

In the first test case, Alice moves the rook to $(2, 1)$ and Bob moves the rook to $(1, 1)$. This process will repeat for 999 times until finally, after Alice moves the rook, Bob cannot move it back to $(1, 1)$ because it has been visited 1000 times before. So the final score of the game is $a_2 + b_1 = 4$.

In the second test case, the final score of the game is $a_3 + b_5$.