D. Iahub and Xors

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

Iahub does not like background stories, so he'll tell you exactly what this problem asks you for.

You are given a matrix a with n rows and n columns. Initially, all values of the matrix are zeros. Both rows and columns are 1-based, that is rows are numbered 1, 2, ..., n and columns are numbered 1, 2, ..., n. Let's denote an element on the i-th row and j-th column as $a_{i,j}$.

We will call a submatrix (x_0, y_0, x_1, y_1) such elements $a_{i,j}$ for which two inequalities hold: $x_0 \le i \le x_1, y_0 \le j \le y_1$.

Write a program to perform two following operations:

1. Query(x_0, y_0, x_1, y_1): print the xor sum of the elements of the submatrix (x_0, y_0, x_1, y_1).
2. Update(x_0, y_0, x_1, y_1, v): each element from submatrix (x_0, y_0, x_1, y_1) gets xor-ed by value v.

Input

The first line contains two integers: n ($1 \le n \le 1000$) and m ($1 \le m \le 10^5$). The number m represents the number of operations you need to perform. Each of the next m lines contains five or six integers, depending on operation type.

If the i-th operation from the input is a query, the first number from i-th line will be 1. It will be followed by four integers x_0, y_0, x_1, y_1. If the i-th operation is an update, the first number from the i-th line will be 2. It will be followed by five integers x_0, y_0, x_1, y_1, v.

It is guaranteed that for each update operation, the following inequality holds: $0 \le v < 2^{62}$. It is guaranteed that for each operation, the following inequalities hold: $1 \le x_0 \le x_1 \le n$, $1 \le y_0 \le y_1 \le n$.

Output

For each query operation, output on a new line the result.

Examples

input
3 5 2 1 1 2 2 1 2 1 3 2 3 2 2 3 1 3 3 3 1 2 2 3 3 1 2 2 3 2

output
3 2

Note

After the first 3 operations, the matrix will look like this:

1 1 2
1 1 2
3 3 3

The fourth operation asks us to compute 1 xor 2 xor 3 xor 3 = 3.

The fifth operation asks us to compute 1 xor 3 = 2.