Codeforces Beta Round #51

A. Flea travel

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

A flea is sitting at one of the n hassocks, arranged in a circle, at the moment. After minute number k the flea jumps through $k - 1$ hassocks (clockwise). For example, after the first minute the flea jumps to the neighboring hassock. You should answer: will the flea visit all the hassocks or not. We assume that flea has infinitely much time for this jumping.

Input

The only line contains single integer: $1 \leq n \leq 1000$ — number of hassocks.

Output

Output "YES" if all the hassocks will be visited and "NO" otherwise.

Sample test(s)

input
1
output
YES

input
3
output
NO

B. Smallest number

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

Recently, Vladimir got bad mark in algebra again. To avoid such unpleasant events in future he decided to train his arithmetic skills. He wrote four integer numbers a, b, c, d on the blackboard. During each of the next three minutes he took two numbers from the blackboard (not necessarily adjacent) and replaced them with their sum or their product. In the end he got one number. Unfortunately, due to the awful memory he forgot that number, but he remembers four original numbers, sequence of the operations and his surprise because of the very small result. Help Vladimir remember the forgotten number: find the smallest number that can be obtained from the original numbers by the given sequence of operations.

Input

First line contains four integers separated by space: $0 \le a, b, c, d \le 1000$ — the original numbers. Second line contains three signs ('+' or '*' each) separated by space — the sequence of the operations in the order of performing. ('+' stands for addition, '*' — multiplication)

Output

Output one integer number — the minimal result which can be obtained.

Please, do not use `%lld` specificator to read or write 64-bit integers in C++. It is preffered to use `cin` (also you may use `%I64d`).

Sample test(s)

input
1 1 1 1 + + *
output
3

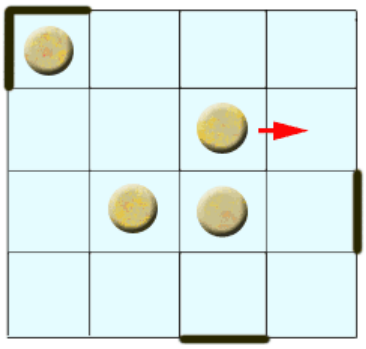
input
2 2 2 2 * * +
output
8

input
1 2 3 4 * + +
output
9

C. Pie or die

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

Volodya and Vlad play the following game. There are k pies at the cells of $n \times m$ board. Each turn Volodya moves one pie to the neighbouring (by side) cell. If the pie lies at the border of the board then Volodya can move it outside the board, get the pie and win. After Volodya's move, Vlad bans some edge at the border of the board of length 1 (between two knots of the board) so that Volodya is not able to move the pie outside the board through this edge anymore. The question is: will Volodya win this game? We suppose both players follow the optimal strategy.



Input

First line contains 3 integers, separated by space: $1 \le n, m \le 100$ — dimensions of the board and $0 \le k \le 100$ — the number of pies. Each of the next k lines contains 2 integers, separated by space: $1 \le x \le n$, $1 \le y \le m$ — coordinates of the corresponding pie. There could be more than one pie at a cell.

Output

Output only one word: "YES" — if Volodya wins, "NO" — otherwise.

Sample test(s)

input
2 2 1 1 2
output
YES

input
3 4 0
output
NO

input
100 50 2 50 25 50 25
output
NO

D. Beautiful numbers

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Volodya is an odd boy and his taste is strange as well. It seems to him that a positive integer number is *beautiful* if and only if it is divisible by each of its nonzero digits. We will not argue with this and just count the quantity of beautiful numbers in given ranges.

Input

The first line of the input contains the number of cases t ($1 \le t \le 10$). Each of the next t lines contains two natural numbers l_i and r_i ($1 \le l_i \le r_i \le 9 \cdot 10^{18}$).

Please, do not use `%lld` specificator to read or write 64-bit integers in C++. It is preffered to use `cin` (also you may use `%I64d`).

Output

Output should contain t numbers — answers to the queries, one number per line — quantities of beautiful numbers in given intervals (from l_i to r_i, inclusively).

Sample test(s)

input
1 1 9
output
9

input
1 12 15
output
2

E. Very simple problem

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

You are given a convex polygon. Count, please, the number of triangles that contain a given point in the plane and their vertices are the vertices of the polygon. It is guaranteed, that the point doesn't lie on the sides and the diagonals of the polygon.

Input

The first line contains integer n — the number of vertices of the polygon ($3 \le n \le 100000$). The polygon description is following: n lines containing coordinates of the vertices in clockwise order (integer x and y not greater than 10^9 by absolute value). It is guaranteed that the given polygon is nondegenerate and convex (no three points lie on the same line).

The next line contains integer t ($1 \le t \le 20$) — the number of points which you should count the answer for. It is followed by t lines with coordinates of the points (integer x and y not greater than 10^9 by absolute value).

Output

The output should contain t integer numbers, each on a separate line, where i-th number is the answer for the i-th point.

Please, do not use `%lld` specificator to read or write 64-bit integers in C++. It is preffered to use `cin` (also you may use `%I64d`).

Sample test(s)

input
4 5 0 0 0 0 5 5 5 1 1 3

output
2

input
3 0 0 0 5 5 0 2 1 1 10 10

output
1 0

input
5 7 6 6 3 4 1 1 2 2 4 4 3 3 2 3 5 5 4 2

output
5 3 3 4