# **A Transpose Matrix**

Given a matrix of size MxN, print the transpose of the matrix.

**Input:**

First line contains two integers M (1<=M<=1000) and N (1<=N<=1000) representing the number of rows and columns of the matrix. Next M line each consist of N 32-bit signed integers.

**Output:**

Transpose of the matrix. See sample for output format.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 2 2  1 2  2 1 | 1 2  2 1 |
| 3 4  1 2 3 4  5 6 7 8  9 1 2 3 | 1 5 9  2 6 1  3 7 2  4 8 3 |

# **B Factorial Table**

Given an integer N, print the Factorial Table for N.

**Input:**

An integer, N (1 ≤ N ≤ 20)

**Output:**

N lines containing factorial calculations from 1 to N. See sample for output format.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 3 | 1! = 1 = 1  2! = 1 × 2 = 2  3! = 1 × 2 × 3 = 6 |
| 5 | 1! = 1 = 1  2! = 1 × 2 = 2  3! = 1 × 2 × 3 = 6  4! = 1 × 2 × 3 × 4 = 24  5! = 1 × 2 × 3 × 4 × 5 = 120 |

# **C SOP/POS Evaluation**

You will be given a boolean expression in either Sum of Product (SOP) or Product of Sum (POS) format and assigned value of all the boolean variables. You have to evaluate the expression depending on the assigned value.

For a given number of variables, names of the variable names are fixed e.g. if there are three variables, they will be A, B and C; if there are four variables, they will be A, B, C and D.

Each SOP/POS term will be enclosed with parentheses if the term has more than one literals. The ‘.’ , ‘+’ and ‘~’ symbols will denote AND, OR and NOT operation respectively. Examples of POS and SOP form boolean expressions with four variables are given below.

POS: (A + B) . (A + ~B + C) . (C + ~D).~A

SOP: (A . B) + (A . ~B . C) + (C . ~D) +A

## Input

An integer N (0<N<10) denoting the number of boolean variables in the expression Followed by N values (0/1) which will be considered as the assigned values for the variables sequentially. Last line of the input will have the boolean expression to be evaluated. There will be no more than 2^10 SOP/POS terms in the expression. Although there might be unnecessary spaces in the expression, the length of the expression string is guaranteed to never exceed 100000.

## Output

A single line stating True or False depending on the evaluated expression.

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| 4  0 1 1 0  (A + B) . (A + ~B + C) . (C + ~D).~A | True |
| 4  0 1 1 1  (A . B) + (A . ~B . C) + (C . ~D) +A | False |

# D Matrix Game

Yeamin loves to play with matrix of integers. But, he doesn’t know how to do different operations on matrix due to his lack of literacy on mathematics. On the otherhand, Noki claimed himself the ultimate boss of the mathematics. As Yeamin is very intelligent, he plans a game to judge the reality about Noki. In this game, Yeamin randomly asked Noki to perform some matrix operations as described below.

Noki has an initial empty (all the elements are zero) matrix of size MxN. In each query, he is asked as:

1. **ADD X Y:** Noki has to add an XxY matrix which is provided by Yeamin with the matrix he already has.
2. **SUB X Y:** Noki has to subtract an XxY matrix which is provided by Yeamin from the matrix he already has.
3. **MUL X Y:** Noki has to multiply an XxY matrix which is provided by Yeamin with the matrix he already has.
4. **ROTATE CLOCK X:** Noki has to rotate X times the matrix he already has in clockwise direction.
5. **ROTATE ANTI X:** Noki has to rotate X times the matrix he already has in anti clockwise direction.

**Input:**

First line contains two integers M (1<=M<=100) and N (1<=N<=100) representing initial empty matrix size that Noki has.

Second line contains T (1<=T<=1000) indicating the number of query from Yeamin. Each of the following query will follow the format described above. All input numbers in this problem is guaranteed to fit in 32-bit signed integer.

**Output:**

Print the matrix that Noki has after each valid query. If Yeamin asked for any invalid query, then print the message “Invalid Operation!”. See sample for output format. Make sure that you have an empty line between two different query outputs. Do not use long long integer to oversome overflow in this problem. In stead of, use 32-bit signed inegers.

**Conditions:**

* 5 test cases will contain only first 3 types of queries: ADD, SUB, MUL
* Rest of the test cases will contain any types of queries described above

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 2 2  3  ADD 2 2  1 2  1 2  SUB 3 2  1 2  1 3  1 4  ROTATE CLOCK 1 | 1 2  1 2  Invalid Operation!  1 1  2 2 |
| 3 3  2  SUB 3 3  1 2 3  5 6 7  9 1 2  MUL 3 3  2 2 2  3 3 3  4 4 4 | -1 -2 -3  -5 -6 -7  -9 -1 -2  -20 -20 -20  -56 -56 -56  -29 -29 -29 |