CCC '05 S3 - Quantum Operations

Time limit: 1.0s
Python 2: 2.5s
Python 3: 2.5s

Memory limit: 256M
Python 2: 64M
Python 3: 64M

Canadian Computing Competition: 2005 Stage 1, Senior #3

Quantum computing is currently a hot topic in research. If they can be built, quantum computers will have the ability to perform certain computing tasks much faster than any computer in existence today. Fortunately, you won't need a quantum computer to do this question.

A fundamental concept in quantum computing is the idea of a quantum operation. A quantum operation can be essentially thought of as a matrix. Also, if you perform two quantum operations in parallel on separate quantum data, this can be thought of as a larger quantum operation. Thinking of these operations in terms of matrices again, the resulting matrix from performing two matrices in parallel is called the *tensor product* of the two matrices (using the symbol $\otimes$). This is different than the normal product of matrices that you may have learned about.

In a tensor product, you are given two matrices (which are essentially two-dimensional arrays). We will call them A and B, and we will represent the individual elements of these two matrices this way:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \ B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1q} \\ b_{21} & b_{22} & \cdots & b_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ b_{p1} & b_{p2} & \cdots & b_{pq} \end{bmatrix}.$$

Notice that the size of matrix A is $m \times n$ (m rows and n columns), and the size of B is $p \times q$.

The tensor product of these two matrices will be an $mp \times nq$ matrix (with mp rows and nq columns) that looks like:

$$A \otimes B = \begin{bmatrix} a_{11}[B] & a_{12}[B] & \cdots & a_{1n}[B] \\ a_{21}[B] & a_{22}[B] & \cdots & a_{2n}[B] \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}[B] & a_{m2}[B] & \cdots & a_{mn}[B] \end{bmatrix},$$

where $a_{ij}[B]$ simply indicates that each element in B is being multiplied by a_{ij}.

Finally notice that the tensor product is not *commutative*, which means that changing the order of matrices may change the answer $(A \otimes B \neq B \otimes A)$.

For more than two matrices, we will define $A \otimes B \otimes C = (A \otimes B) \otimes C$, although it does not technically matter, since the tensor product is *associative*.

Your task is to compute and output the tensor product of two or more given matrices.

Input Specification

The first line of input will contain the number of matrices, N, a positive integer. Then, there are N blocks of lines describing the matrices in order.

In each block, the first line will contain two positive integers, r and c, separated by a space, indicating the number of rows and columns, respectively. Then, the next r lines will denote the rows, in order. Each line will contain c integers, separated by spaces.

Output Specification

The output (to the screen) will be 6 integers in the following order:

- the maximum element in the tensor product
- the minimum element in the tensor product
- the maximum row sum (i.e., sum of entries in each row)
- the minimum row sum
- the maximum column sum
- the minimum column sum

You may assume that the tensor product matrix will have no more than 1024 rows and no more than 1024 columns.

Sample Input 1

```
2
2 2
1 1
1 -1
2 2
1 0
0 1
```

Sample Output 1

```
1
-1
2
0
2
0
```

Tensor Product for Sample Input 1

```
1 0 1 0
0 1 0 1
1 0 -1 0
0 1 0 -1
```

Sample Input 2

```
3
2 2
1 0
0 3
2 2
1 1
1 -1
2 2
1 0
0 1
```

Sample Output 2

```
3
-3
6
0
6
0
```

Tensor Product for Sample Input 2

```
1 0 1 0 0 0 0 0
0 1 0 1 0 0 0 0
1 0 -1 0 0 0 0 0
0 1 0 -1 0 0 0 0
0 0 0 0 3 0 3 0
0 0 0 0 0 3 0 3
0 0 0 0 3 0 -3 0
0 0 0 0 0 3 0 -3
```