Lab: Multidimensional Lists

This document defines the exercises for the "Python Advanced" course at @Software University.
Please submit your solutions (source code) to all the below-described problems in Judge.

1. Sum Matrix Elements

Write a program that **reads a matrix** from the console and prints:

- The **sum** of all **numbers** in the matrix
- The **matrix itself**

On the first line, you will receive the matrix sizes in the format **`"{rows}, {columns}"`.** On the next **rows**, you will get elements for each column separated by a comma and a space **`", "`**.

Examples

Input	Output
3, 6 7, 1, 3, 3, 2, 1 1, 3, 9, 8, 5, 6 4, 6, 7, 9, 1, 0	76 [[7, 1, 3, 3, 2, 1], [1, 3, 9, 8, 5, 6], [4, 6, 7, 9, 1, 0]]

2. Even Matrix

Write a program that receives a **matrix of numbers** and prints a **new one** only with the even numbers. Use nested comprehension for that problem.

On the first line, you will receive **the rows of the matrix**. On the next **rows**, you will get elements for each column separated with a comma and a space **`", "`**.

Examples

Input	Output
2 1, 2, 3 4, 5, 6	[[2], [4, 6]]
4 10, 33, 24, 5, 1 67, 34, 11, 110, 3 4, 12, 33, 63, 21 557, 45, 23, 55, 67	[[10, 24], [34, 110], [4, 12], []]

3. Flattening Matrix

Write a program that receives a **matrix** and prints its flattened version (a list with all the values). For example, the flattened list of the matrix: **`[[1, 2], [3, 4]]`** will be **`[1, 2, 3, 4]`**.

On the first line, you will receive the **number of a matrix's rows.** On the next **rows**, you will get the **elements** for **each column** separated with a comma and a space **`", "`**.

Examples

Input	Output
2 1, 2, 3 4, 5, 6	[1, 2, 3, 4, 5, 6]
3 10, 2, 21, 4 5, 20, 41, 9 6, 2, 4, 99	[10, 2, 21, 4, 5, 20, 41, 9, 6, 2, 4, 99]

4. Sum Matrix Columns

Write a program that **reads a matrix** from the console and prints the **sum for each column** on separate lines.

On the first line, you will get matrix sizes in the format **"{rows}, {columns}"**. On the next **rows**, you will get **elements** for **each column** separated with a single space.

Examples

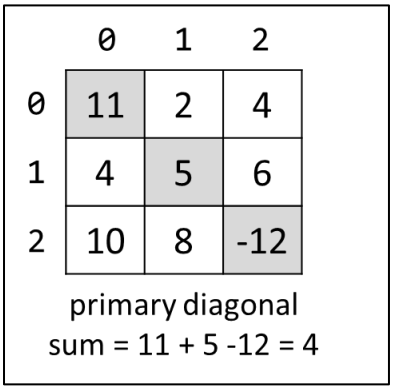
Input	Output	Input	Output
3, 6 7 1 3 3 2 1 1 3 9 8 5 6 4 6 7 9 1 0	12 10 19 20 8 7	3, 3 1 2 3 4 5 6 7 8 9	12 15 18

Hints

- **Read** matrix **sizes**.
- On the next row lines, **read** the **columns**.
- **Traverse** the matrix and **sum** all elements in **each** column.
- Print the **sum** and **continue** with the other columns.

5. Primary Diagonal

Write a program that finds the **sum of all numbers in a matrix's primary diagonal** (runs from top left to bottom right). On the **first line**, you will receive an integer **N** – the size of a square matrix. The next **N lines** hold the values for **each column** - **N** numbers, separated by a single space.



Examples

Input	Output	Input	Output
3 11 2 4 4 5 6 10 8 -12	4	3 1 2 3 4 5 6 7 8 9	15

6. Symbol in Matrix

Write a program that reads a number - **N**, representing the **rows** and **columns** of a square **matrix**. On the next **N** lines, you will receive rows of the matrix. Each row consists of ASCII characters. After that, you will receive a symbol. Find the **first occurrence** of that symbol in the matrix and print **its position** in the format: "**`({row}, {col})`**". It would help if you started searching from the **top left**. If there is no such symbol, print the message "**`{symbol} does not occur in the matrix`**".

Examples

Input	Output
3 ABC DEF X!@ !	(2, 1)
4 asdd xczc qwee qefw 4	4 does not occur in the matrix

7. Square with Maximum Sum

Write a program that **reads a matrix** from the console and finds the **2x2 top-left submatrix** with the **biggest sum** of its values.

On the first line, you will get matrix sizes in the format **`"{rows}, {columns}".`** On the next **rows**, you will get **elements** for **each column**, separated with a comma and a space **`", "`**.

You should print the **found submatrix** and the **sum of its elements,** as shown in the examples.

Examples

Input	Output
3, 6 7, 1, 3, 3, 2, 1 1, 3, 9, 8, 5, 6 4, 6, 7, 9, 1, 0	9 8 7 9 33

2, 4 10, 11, 12, 13 14, 15, 16, 17	12 13 16 17 58

Hints

- Be aware of **`IndexError`**
- If you find more than one max square, print the **top-left one**