

Lab: Multidimensional Arrays

Problems for exercises and homework for the ["CSharp Advanced" course @ Software University](#).

You can check your solutions here: <https://judge.softuni.bg/Contests/1452/Multidimensional-Arrays-Lab>

1. Sum Matrix Elements

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

- Count of **rows**
- Count of **columns**
- Sum of all **matrix elements**

On first line you will get matrix sizes in format **[rows, columns]**

Examples

| Input | Output |
|------------------|--------|
| 3, 6 | |
| 7, 1, 3, 3, 2, 1 | 3 |
| 1, 3, 9, 8, 5, 6 | 6 |
| 4, 6, 7, 9, 1, 0 | 76 |

Hints

- On next **[rows]** lines you will get elements for each column separated with coma and whitespace
- Try to use only **foreach** for printing

2. Sum Matrix Columns

Write program that **read a matrix** from console and print the sum for each column. On first line you will get matrix **rows**. On the next **rows** lines, you will get elements for each column separated with a space.

Examples

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

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.

3. Primary Diagonal

Write a program that finds the **sum of matrix primary diagonal**.

| | 0 | 1 | 2 |
|---|----|---|-----|
| 0 | 11 | 2 | 4 |
| 1 | 4 | 5 | 6 |
| 2 | 10 | 8 | -12 |

primary diagonal
sum = 11 + 5 -12 = 4

Input

- On the **first line**, you are given the integer **N** – the size of the square matrix
- The next **N lines** holds the values for **every row** – **N** numbers separated by a space

Examples

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

4. Symbol in Matrix

Write a program that reads **N**, number representing **rows** and **cols** of a **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}**". If there is no such symbol print an error 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 |

5. Square with Maximum Sum

Write a program that **read a matrix** from console. Then find biggest sum of **2x2 submatrix** and print it to console.

On first line you will get matrix sizes in format **rows, columns**.

One next **rows** lines you will get elements for each **column** separated with coma.

Print **biggest top-left** square, which you find and sum of its elements.

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

- Think about `IndexOutOfRangeException()`
- If you find more than one max square, print the top-left one

6. Jagged-Array Modification

Write a program that **reads a matrix** from the console. On the first line you will get matrix **rows**. On next **rows** lines you will get elements for each **column** separated with **space**. You will be receiving commands in the following format:

- **Add {row} {col} {value}** – Increase the number at the given **coordinates** with the **value**.
- **Subtract {row} {col} {value}** – Decrease the number at the given **coordinates** by the **value**.

Coordinates might be invalid. In this case you should print **"Invalid coordinates"**. When you receive **"END"** you should print the matrix and stop the program.

Examples

| Input | Output |
|---|-------------------------|
| 3 1 2 3 4 5 6 7 8 9 Add 0 0 5 | 6 2 3 4 3 6 7 8 9 |

| | |
|--|--|
| Subtract 1 1 2 END | |
| 4 1 2 3 4 5 6 7 8 8 7 6 5 4 3 2 1 Add 4 4 100 Add 3 3 100 Subtract -1 -1 42 Subtract 0 0 42 END | Invalid coordinates Invalid coordinates -41 2 3 4 5 6 7 8 8 7 6 5 4 3 2 101 |

7. Pascal Triangle

The triangle may be constructed in the following manner: In row 0 (the topmost row), there is a unique nonzero entry 1. Each entry of each subsequent row is constructed by adding the number above and to the left with the number above and to the right, treating blank entries as 0. For example, the initial number in the first (or any other) row is 1 (the sum of 0 and 1), whereas the numbers 1 and 3 in the third row are added to produce the number 4 in the fourth row.

If you want more info about it: https://en.wikipedia.org/wiki/Pascal's_triangle

Print each row elements separated with whitespace.

Examples

| Input | Output |
|-------|---|
| 4 | 1 1 1 1 2 1 1 3 3 1 |
| 13 | 1 1 1 1 2 1 1 3 3 1 1 4 6 4 1 1 5 10 10 5 1 1 6 15 20 15 6 1 1 7 21 35 35 21 7 1 1 8 28 56 70 56 28 8 1 1 9 36 84 126 126 84 36 9 1 1 10 45 120 210 252 210 120 45 10 1 1 11 55 165 330 462 462 330 165 55 11 1 1 12 66 220 495 792 924 792 495 220 66 12 1 |

Hints

- The input number **n** will be **1 ≤ n ≤ 60**
- Think about proper **type** for elements in array
- Don't be scary to use **more and more arrays**