# Упражнения: Многомерни масиви

Тествайте решението в Judge: <https://judge.softuni.org/Contests/4156/04-Multidimentional-Arrays-Basics>

## Основен диагонал

Напишете програма, която намира **сумата от основните диагонали на матрицата**. Основен диагонал на матрица представлява диагонала от **всички елементи**, чиито индекси за **номер на** **ред** и **номер на** **колона** са **равни** – ред 0, колона 0 – ред 1, колона 1 – ред 2, колона 2 и т.н.

Graphical user interface, application, Teams

Description automatically generated

### Вход

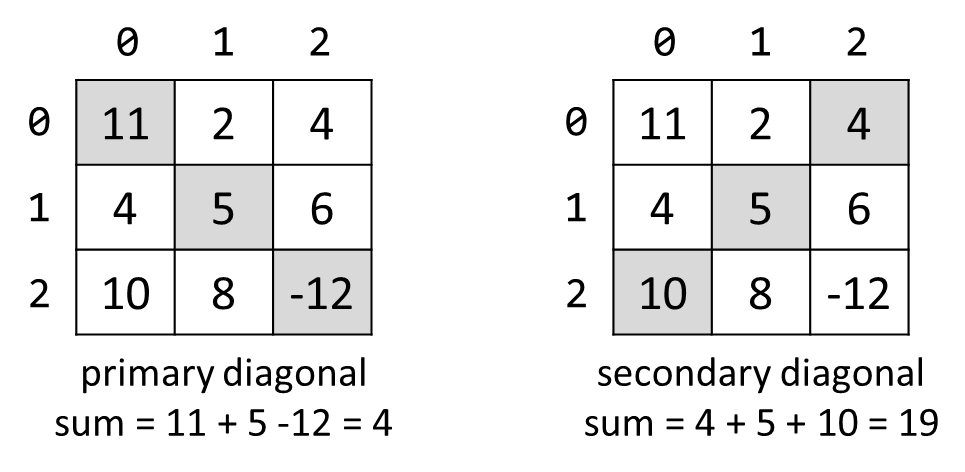
* На **първия ред** ще получите числото **N** - размер на квадратната матрицата
* На следващите **N реда** ще получите стойностите на **всички редове** - **N** числа разделени с интервал

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 3  **11** 2 4  4 **5** 6  10 8 **-12** | 4 |
| 3  **1** 2 3  4 **5** 6  7 8 **9** | 15 |

## Разлика на диагонали

Създайте програма, която намира **разликата на сумите от диагоналите** (абсолютната стойност).



### Вход

* На **първия ред** ще получите числото N - **размера** на квадратната матрица
* На следващите **N реда** ще получавате стойностите на **всички редове** - **N** числа, разделени с интервал

### Изход

* Отпечатайте **абсолютната** стойност на разликата между основния и втория диагонал

### Примери

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснение** |
| 3  11 2 4  4 5 6  10 8 -12 | 15 | **Основния диагонал:** sum = 11 + 5 + (-12) = 4  **Втория диагонал:** sum = 4 + 5 + 10 = 19  **Разлика:** |4 - 19| = 15 |

## Сума на елементите от матрица

Напишете програма, която **чете** от входа матрица и **отпечатва**:

* Броя на **редовете**
* Броя на **колоните**
* Сумите на всички **елементи от матрицата**

На първия ред ще получите **размерите на матрицата** в следния формат: **“ред, колона”**. На следващите редове ще прочетете **клетките** на матрицата.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 3, 6  7, 1, 3, 3, 2, 1 1, 3, 9, 8, 5, 6 4, 6, 7, 9, 1, 0 | 3  6  76 |

### Насоки

* Опитайте се да използвате **foreach-цикъл**

## Сума на колоните на матрица

Напишете програма, която **чете матрица** от входа на конзолата и **отпечатва сумата на всички колони**. На първия ред ще получите **размерите** на матрицата в следния формат: **"редове , колони",** разделени с запетая и интервал. На следващите редовеще получите елементите за **всяка колона** разделени с интервал.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 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 |

### Насоки

* Прочете размера на матрицата.
* На следващите редове ще получите колоните.
* Обходете матрицата и съберете сумата на всички елементи във всяка колона.
* Отпечатайте сумата и продължете с другите колони.

## **Квадрат с най-голяма сума**

Напишете програма, която **чете матрица** от конзолата. Трябва да намерите сумата от квадрат с **дължина 2x2** от матрицата и да го **отпечатате**.

На първия ред ще получите **размерите** в следния формат **"редове, колони".**

На следващите редовеще получите елементите за **всяка** **колона**,разделени с запетая и интервал .

Отпечатайте **най-големия** квадрат и неговата сума.

### Примери

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 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 | 7, 1, 3, 3, 2, 1 1, 3, **9, 8**, 5, 6 4, 6, **7, 9**, 1, 0 |
| 2, 4  10, 11, 12, 13  14, 15, 16, 17 | 12 13  16 17  58 | 10, 11, **12, 13**  14, 15, **16, 17** |

### Насоки

* Помислете за **IndexOutOfRangeException()**
* Ако намерете повече от един квадрат с най-голяма сума, отпечатайте този най-горе вляво.

## **Модификация на назъбен масив**

Напишете програма, която **чете матрица** от конзолата и **чете команди**. На първия ред ще получите **редовете**. На следващите редовеще получите елементите за **всяка** **колона**,разделени с **интервал**. Командите ще бъдат в следния формат:

* **Add {ред} {колона} {стойност}** – **Увеличава** числото на дадените **координати** със **стойността.**
* **Subtract {ред} {колона} {стойност}** – **Намалява** числото на дадените **координати** със **стойността.**

Координатите може и да **не бъдат** валидни. В такъв случай отпечатайте "**Invalid coordinates**". Когато получите "**END**", програмата спира и трябва да отпечатате матрицата.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 3  **1** 2 3  4 5 **6**  7 8 9  Add 0 0 5  Subtract 1 2 1  END | **6** 2 3  4 **5** **5**  7 8 9 |
| 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 |

## **Триъгълника на Паскал**

Триъгълникът може да бъде създаден по следния начин: В **редица 0** (най-горната) има **уникално число** със стойност **1**. Всяка стойност от всяка следваща редица се създава чрез **събиране** на числото **отгоре отляво** с числото **отгоре отдясно**, третирайки празните места с **0**. Например първоначалното число в първата (или която и да е друга) редица е 1 (сумата от 0 и 1), докато числата 1 и 3 в третата редица се добавят, за да се получи числото 4 в четвъртата редица.

За повече информация научете на: [https://bg.wikipedia.org/wiki/Триъгълник\_на\_Паскал](https://bg.wikipedia.org/wiki/%D0%A2%D1%80%D0%B8%D1%8A%D0%B3%D1%8A%D0%BB%D0%BD%D0%B8%D0%BA_%D0%BD%D0%B0_%D0%9F%D0%B0%D1%81%D0%BA%D0%B0%D0%BB)

Отпечатайте **елементите** на всяка редица, разделени с **интервал.**

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 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 |

### Насоки

* Входа **n** винаги ще бъде: **1 <= n <= 60**
* Помислете от какъв **тип данни** ще бъде масива
* Не се страхувайте да **използвате повече от един масив**