# Упражнения: Многомерни масиви

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

## Отпечатване на матрица

Напишете програма, която първо прочита **цяло число** **n**. След това ще прочетете цели числа, които да запишете в **nxn** **матрица**. Числата ще се въвеждат по **едно на ред**. След това отпечатайте **матрицата** (елементите на всеки ред трябва да бъдат разделени с **интервал**), както и **сумата на числата в нея**.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| **2**  3  4  6  5 | 3 4  6 5  Sum: 18 |
| 3  1  2  3  4  5  6  7  8  9 | 1 2 3  4 5 6  7 8 9  Sum: 45 |

## Основен диагонал

Напишете програма, която прочита число **n** , след което чете редовете на **nxn матрица**. Програмата трябва да намира **сумата от основните диагонали на двумерна матрица**. 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

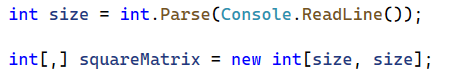
* На **първия ред** ще получите числото **n** - размер на квадратната матрицата
* На следващите **n реда** ще получите стойностите на **всички редове** - **n** числа разделени с интервал

### 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 |

### Насоки

1. Прочетете **число** от конзолата, което да определи **размера** на **матрицата**, и я **създайте**:



1. Създайте **променилива**, която да пази **сумата** на **диагонала**:



1. Направете **for-цикъл**, чрез който да **прочетете** данни за **всеки ред**:

A computer screen shot of a code

Description automatically generated

1. **Вложете** **for-цикъл**, който да **попълва** данни в **текущия** **ред** на определена **колона** в **матрицата**:

A math equation with black text

Description automatically generated with medium confidence

1. В този **for-цикъл** направете **проверка** дали **реда е равен на колоната** и ако **е**, **прибавете** **текущата** **стойност** към **крайната сума**:

A black and blue text

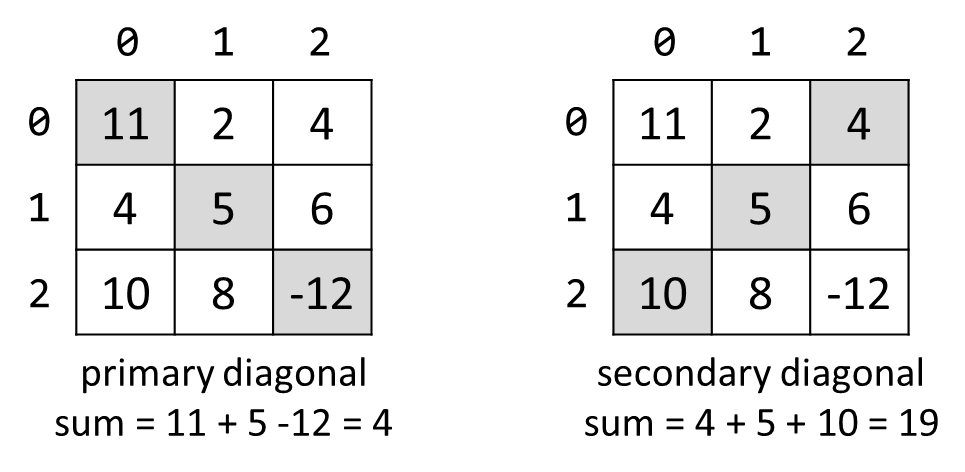
Description automatically generated

1. **Принтирайте** сумата:



## Разлика на диагонали

Създайте програма, която чете **nxn матрица** и 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 |

### Насоки

1. Прочетете входните данни – размера на матрицата (**size**):

A close up of words

Description automatically generated

1. Създайте матрицата и **попълнете** данни в нея чрез метода **FillTheMatrix(int[,] matrix)**, който ще имплементирате след малко.:

A close up of a text

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Имплементирайте методите **GetsPrimarySum(int[,] matrix)** и **GetsSecondarySum(int[,] matrix)**:

* Чрез **GetsPrimarySum(int[,] matrix)** взимаме **сумата** за **главния диагонал** по следния начин:

A screenshot of a computer program

Description automatically generated

* Чрез **GetsSecondarySum(int[,] matrix)** взимаме **сумата** за **противоположния диагонал** по следния начин:

A screen shot of a computer program

Description automatically generated

1. Създаваме **променливи**, които да пазят **сумите** на **двата диагонала**:

A black and orange text

Description automatically generated with medium confidence

1. Намираме **абсолютната разлика** на **сумите** и я **принтираме** на **конзолата**:



## 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 | 7 1 3 3 2 1  1 3 9 8 5 6  4 6 7 9 1 0  Rows: 3  Columns: 6  Sum of elements: 76 |

### Насоки

1. Прочетете **параметри** за създаване матрицата (**брой** на **редове** и **колони**):

A close-up of a computer screen

Description automatically generated

1. Създайте матрицата и направете **променлива**, пазеща **сумата**:



1. Използвайте **for-цикъл** и за всеки ред **прочетете** **данни** от **конзолата**, след което ги **вмъкнете** в **матрицата**. Добавете ги към **сумата**:

A close-up of a computer screen

Description automatically generated

1. **Принтирайте** данните в **зададения формат**.

## 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 със **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

1. Прочете **размерите** на матрицата:

A close-up of a computer screen

Description automatically generated

1. На следващите редове ще получите **колоните**.

A computer screen shot of a program code

Description automatically generated

1. Обходете матрицата и съберете **сумата** на всички елементи във всяка колона.
2. 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

1. Прочете **размерите** на матрицата:
2. **Попълнете** матрицата с входни данни.
3. Обиколете матрицата и вземете **сумата** на **текущия** **квадрат**, след което **проверете** дали тя е **по-голяма** от **текущата**:

* Помислете за **IndexOutOfRangeException()**

1. **Отпечтайте** резултата на конзолата.

* Ако намерете **повече от един** квадрат с **най-голяма сума**, отпечатайте **първия** такъв.

## **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 |

### Бележки

* Входът **n** винаги ще бъде: **1 <= n <= 60**