# Упражнения: Многомерни масиви

Тествайте решението в Judge: <https://judge.softuni.org/Contests/4157/04-Multidimentional-Arrays-Advanced>

## 2x2 Квадрат

Намерете броя на **квадратите** от **еднакви букви** с дължина **2x2** в дадена матрица.

### Вход

* На първия редще ви бъде даден броя на **редовете** и броя на **колоните** - измеренията на матриците
* На следващите редове ще ви бъдат дадени **редовете** на матриците.

### Изход

* Трябва да отпечатате **броя** на еднаквите 2x2 квадрати от матрицата.

### Примери

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 3 4  A B B D  E B B B  I J B B | 2 | Две 2x2 квадрата с еднакви клетки:  A **B B** D A B B D  E **B B** B E B **B B**  I J B B I J **B B** |
| 2 2  a b  c d | 0 | Няма съществуващи 2x2 квадрата с еднакви клетки. |

## Максимална сума 3x3

Напишете програма, която чете правоъгълна матрица с размери **N** и **M**. Трябва да намерите **квадрата** с най-голяма сума, с **размери 3x3**.

### Вход

* На първия ред ще получите **N** реда и **M** колони
* На следващите **N** реда ще получите **редовете на матрицата**

### Изход

* Отпечатайте **елементите** на квадрата от матрицата с дължина 3x3 и тяхната **сума**.

### Примери

|  |  |  |
| --- | --- | --- |
| **Вход** | **Матрица** | **Изход** |
| 4 5  1 5 5 2 4  2 1 4 14 3  3 7 11 2 8  4 8 12 16 4 |  | Sum = 75  1 4 14  7 11 2  8 12 16 |

## Символи в матрицата

Напишете програма, която чете число **N.** Числото представлява **колоните** и **редовете** на **матрицата**. На следващите **N реда** ще получите **редовете на матрицата**. Всяка редица ще съдържа **ASCII символи**. Накрая ще получите **символ**. Намерете първия символ, който **съвпада** с него и отпечатайте неговата позиция в следния формат: "({редица}, {колона})". Ако няма съвпадения, отпечатайте: "{символът} does not occur in the matrix ".

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 3  ABC  DEF  X**!**@  ! | (2, 1) |
| 4  asdd  xczc  qwee  qefw  4 | 4 does not occur in the matrix |

## Змия

Разхождате се в парка. Срещу вас стои змия. Уплашени, започвате да бягате на зиг-заг. Змията започва да ви следва.

Вашата задача е да визуализирате пътя на змията в квадратна форма. **Змията** е представена като **низ**. Движи се по матрица с дължина **NxM.** Змията започва от **горният ляв ъгъл** като се плъзга надолу. **Първата клетка** е запълнена с **първия символ** на змията, **втората клетка** е запълнена с **втория символ** на змията и т.н. Змията е толкова дълга, **колкото е необходимо**, за да запълни реда изцяло – ако стигнете до края на низа, змията започва отново от началото. След като попълните матрицата с пътя на змията, трябва да я отпечатате.

### Вход

* На първия ред ще получите **измеренията** в следния формат: **"N M"**, където **N** са **редовете**, а **M** са **колоните**. Ще ги получите разделени с интервал.
* На втория ред ще получите низ - **змията**

### Изход

* Изходът трябва да отпечата на конзолата **N реда.**
* Всеки ред трябва да съдържа низ, който представлява съответния ред от матрицата

### Бележки

* **Измеренията** N и M ще бъдат в обхвата [1 … 12]
* **Змията** ще бъде низ с дължина [1 … 20] и **няма да съдържа интервал**

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5 6  SoftUni | SoftUn  UtfoSi  niSoft  foSinU  tUniSo |

## Манипулиране на назъбен масив

Създайте програма, която манипулира елементите от матрица с различни дължини на редовете.

Първо прочетете **числото N**, което е равно на **номера на редовете** в матрицата.

На **следващите N редове** ще ви бъде **дадени редове от числа (разделена с интервал) на матрицата**.

След попълването на матрицата, започнете да я анализирате. Ако сегашната редицата и редицата под нея имат **еднаква дължина**, **умножете** всякочисло по **2**. В **противен** случай ги **разделете** на **2**.

След това ще получите командите. Има три възможни команди:

* "**Add {ред} {колона} {стойност}**" - Добавя **{стойност}** на елемента на дадени индекси, ако индексите са валидни
* "**Subtract {ред} {колона} {стойност}**" - Изважда **{стойност}** от елемента на дадени индекси, ако са валидни
* "**End**" - принтира крайната матрица (всички елементи са разделени с интервал) и спира програмата

### Вход

* На първия ред ще получите **броя на редовете** на матрицата - число **N**
* На следващите **N редове** ще получите **редовете на матрицата** - редици от числа, разделени с интервал
* Ще получавате команди докато не получите **"End"**

### Изход

* Изходът трябва да бъде принтиран на конзолата. Трябва да съдържа **N реда**.
* Всеки ред трябва да съдържа низ - **редица** на матрицата. Елементите на всяка редица трябва да бъдат разделени с **интервал**

### Бележки

* **Броя** на редовете е число в обхвата на [2 … 12]
* **Входа** винаги ще бъде във **формата** по-горе
* **Помислете** какъв тип данни ще използвате

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5  10 20 30  1 2 3  2  2  10 10  End | 20 40 60 1 2 3 2 2 5 5 |
| 5  10 20 30  1 2 3  2  2  10 10  Add 0 10 10  Add 0 0 10  Subtract -3 0 10  Subtract 3 0 10  End | 30 40 60  1 2 3  2  -8  5 5 |

## Разбъркана матрица

Напишете програма, която чете от входа матрица от низове и модифицира нейните елементи. Въвеждането от потребителя представлява вход както на задачите по-горе – първо четете **измеренията** и след това **данните**.

Вашата програма трябва да получи команди в следния формат: "**swap редица1 колона1 редица2 колона2**", където редица1, колона1, редица2, колона2 са координати в матрицата. Трябва да сменяте стойностите на дадените координати (клетка [редица1, колона1]) с клетка[редица2, колона2]) и трябва да отпечатате всяка стъпка (така ще можете да проверите дали така е вярно изпълнена).

Ако командата е **невалидна** (няма дума "**swap**", има повече координати или координатите са невалидни) отпечатайте "**Invalid input!**" и преминете към следващата команда. Вашата програма спира, когато получите командата "**END**".

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 2 3  1 2 3  4 5 6  swap 0 0 1 1  swap 10 9 8 7  swap 0 1 1 0  END | 5 2 3  4 1 6  Invalid input!  5 4 3  2 1 6 |

|  |  |
| --- | --- |
| 1 2  Hello World  0 0 0 1  swap 0 0 0 1  swap 0 1 0 0  END | Invalid input!  World Hello  Hello World |