# Exercises: Arrays: Simple Array Processing

Problems for exercises and homework for the [“Programming Fundamentals” course @ SoftUni](https://softuni.bg/courses/programming-fundamentals).

You can check your solutions here: <https://judge.softuni.bg/Contests/421>.

## Largest Element in Array

Read an integer **N** and on the next **N** lines read an **array** of **integers.** Then, find its **largest** element.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  5  6  8  7 | 8 |
| 3  -2  -18  -5348 | -2 |
| 6  2  2  2  2  2  2 | 2 |

## Count of Negative Elements in Array

Read an integer **N** and on the next **N** lines read an **array** of **integers.** Then, find the **count** of **negative** elements in the array.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  1  -2  3  -4  5 | 2 |
| 3  1  3  2 | 0 |
| 6  -1  -2  -3  -4  -4  4 | 5 |

## Count of Given Element in Array

Read an array of integers and print how many times a given element exists in it.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 2 3 1 5  2 | 1 |
| 1 4 4 4 1  4 | 3 |
| 8 4 9 0 0  0 | 2 |

### Hints

* Use the .Split() function to read the array on a single line.

## Count Occurrences of Larger Numbers in Array

Read an **array** of **real numbers** and a number **p**. Find how many elements are **bigger** than **p** in the array and **print** the count.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3 2 3.5 3 2 4 3 4 -2 -7 3  2.9 | 7 |
| 5 6 105 3 2 849  100 | 2 |
| 1.5 4.1 9.3 10.5 0.85  4.9 | 2 |

## Increasing Sequence of Elements

Read an **array of integers** and find out if it is an **increasing sequence**. Print Yes if it does and No if it doesn’t.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 5 10 12 | Yes |
| 1 5 2 12 | No |

## Equal Sequence of Elements in Array

Read an **array of integers** and find out if all the elements in the array are **the same**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 5 5 | Yes |
| 3 4 4 | No |

## Count of Capital Letters in Array

Read an **array of strings** and find out how many of them are **capital English letters** (such as A, B, C etc.). Print the count on the console.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Hello SoftUni I am A student | 2 |
| I Think A B and C are the first three letters of the alphabet | 4 |

## Array Symmetry

Read an **array of strings** and find out if it’s **symmetric**. If it is, print “Yes”, otherwise print “No”.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| hi you hi | Yes |
| ho hi hi ho | Yes |
| hi ho ho ho | No |

### Hints

* Iterate over the array as follows:
  + Check whether the **first** and **last** elements are equal
  + Then check whether the **second** and the **next to last** elements are equal
  + Continue this pattern until you either **find an inequality** or **reach the middle** of the array.
* Alternate solution: **reverse** the array and check if it is the **same** as the **original** array.

## \* Altitude

You are an airplane pilot, trying to maneuver your airplane to safety from an unknown danger.

An array holds a sequence of up / down commands and **numbers**. Its first element **always** holds the **initial altitude**. Thecommand up **increases** the altitude by the next number, while the command **down decreases** the altitude by the **next number**.

If at any point the altitude becomes either **zero** or **negative**, print “crashed” and **end** **the program**. If by the end, the altitude is **positive**, however, print “got through safely. current altitude: {altitude}m”.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 300 up 3 down 7 up 5 | got through safely. current altitude: 301m |
| 50 up 2 down 20 up 3 down 36 up 1 down 2 up 2 | crashed |
| 11 up 12 down 20 down 3 down 36 down 1 down 2 down 2 | crashed |

## \* Ballistics Training

You are the anti-aircraft operator, trying to shoot down the airplane from the previous problem. You’ll be given instructions to get to the current coordinates of the plane. Shoot it down.

You will be given an array which holds 2 numbers – the **target** **X** and **Y** **coordinates** of the plane.

Afterwards, you will be given a second array, which holds a sequence of **left** / **right** / **up** / **down** commands and **numbers**. We start at position **{x=0, y=0}**. Manipulate the firing position in the following way:

* Up **increases** **y** by the next number.
* Down **decreases** **y** by the next number.
* Left **decreases x** by the next number.
* Right **increases x** by the next number.

After you process all the commands, print “firing at [{x}, {y}]”. After that, check if the **firing position** **coordinates** line up with the **target coordinates**. If they do, print “got 'em!”. If not – print “better luck next time...”

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 25 -3  right 18 down 6 left -7 up 3 | firing at [25, -3]  got 'em! |
| 150 33  right 108 down 4 left -11 up 3 right 30 up 33 right 2 | firing at [151, 32]  better luck next time... |