

Lab: Arrays

Problems for in-class lab for the ["JavaScript Advanced" course @ SoftUni](https://judge.softuni.bg/Contests/2752/Arrays-and-Nested-Arrays-Lab). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/2752/Arrays-and-Nested-Arrays-Lab>.

Arrays

1. Even Position Element

Write a function that finds the elements at even positions in an array.

The **input** comes as an **array of string** elements.

The **output** is printed on the console. Collect all elements in a string, separated by space.

Examples

Input	Output
['20', '30', '40', '50', '60']	20 40 60

Input	Output
['5', '10']	5

2. Last K Numbers Sequence

You are given two integers **n** and **k**. Write a JS function that generates and **return** the following sequence:

- The first element is 1
- Every following element equals the **sum** of the previous **k** elements
- The length of the sequence is **n** elements

The **input** comes as **two number arguments**. The first element represents the number **n**, and the second – the number **k**.

The **output** is the **return** value of your function and should be an **array of numbers**.

Example

Input	Output
6, 3	[1, 1, 2, 4, 7, 13]

Input	Output
8, 2	[1, 1, 2, 3, 5, 8, 13, 21]

Explanation

The 2nd element (1) is the sum of the 3 elements before it, but there is only 1, so we take that. The third element is the sum of the first 2 (1 and 1) and the 4th – the sum of 1, 1 and 2. The 5th element is the sum of the 2nd, 3rd and 4th (1, 2 and 4) and so on.

3. Sum First Last

Write a function that calculates and return the sum of the first and the last elements in an array.

The **input** comes as an **array of string elements** holding numbers.

The **output** is the **return** value of your function and should be a **number**.

Example

Input	Output
['20', '30', '40']	60

Input	Output
['5', '10']	15

4. Negative / Positive Numbers

Write a JS function that processes the elements in an array one by one and produces a new array. **Prepend** each **negative** element at the front of the result and **append** each **positive** (or 0) element at the end of the result.

The **input** comes as an **array of number elements**.

The **output** is printed on the console, each element on a new line.

Example

Input	Output
[7, -2, 8, 9]	-2 7 8 9

Input	Output
[3, -2, 0, -1]	-1 -2 3 0

5. Smallest Two Numbers

Write a function that prints the two smallest elements from an array of numbers.

The **input** comes as an **array of number elements**.

The **output** is printed on the console on a single line, separated by space.

Example

Input	Output
[30, 15, 50, 5]	5 15

Input	Output
[3, 0, 10, 4, 7, 3]	0 3

6. Bigger Half

You are given an array of numbers. Write a JS function that **sorts** the array in **ascending order** and returns a new array, containing only the **second half** of the input. If there is an odd number of elements in the input, always take the bigger half. For example, if the input array contains 4 elements, the output should be 2, and if the input is 5 – the output is 3.

The **input** comes as an **array of number elements**.

The **output** is the **return** value of the function and should be an **array of numbers**.

Example

Input	Output
[4, 7, 2, 5]	[5, 7]
[3, 19, 14, 7, 2, 19, 6]	[7, 14, 19, 19]

7. Piece of Pie

Write a function that receives **three parameters** – an **array** of pie flavors as **strings**, two target flavors as **strings**. The result of the function should be a **new array**, containing a section of the original array, **starting** at the first flavor parameter, and **ending** at (and **including**) the second flavor parameter.

The **input** comes as **three arguments**:

- An **array of strings**, representing pie flavors
- **Two more strings**, representing the start and end of the section, respectively

The **output** is the **return** value of the function and should be an **array of strings**.

Example

Input	Output
<pre>['Pumpkin Pie', 'Key Lime Pie', 'Cherry Pie', 'Lemon Meringue Pie', 'Sugar Cream Pie'], 'Key Lime Pie', 'Lemon Meringue Pie'</pre>	<pre>['Key Lime Pie', 'Cherry Pie', 'Lemon Meringue Pie']</pre>
<pre>['Apple Crisp', 'Mississippi Mud Pie', 'Pot Pie', 'Steak and Cheese Pie', 'Butter Chicken Pie', 'Smoked Fish Pie'], 'Pot Pie', 'Smoked Fish Pie'</pre>	<pre>['Pot Pie', 'Steak and Cheese Pie', 'Butter Chicken Pie', 'Smoked Fish Pie']</pre>

8. Process Odd Positions

You are given an array of numbers. Write a JS function that **return** the elements at **odd positions** from the array, **doubled** and in **reverse** order.

The **input** comes as an **array of number elements**.

The **output** is **return** on the console on a single line, separated by space.

Example

Input	Output
[10, 15, 20, 25]	50 30

Input	Output
[3, 0, 10, 4, 7, 3]	6 8 0

Nested Arrays

9. Biggest Element

Write a function that finds the biggest element inside a matrix.

The **input** comes as an **array of arrays**, containing number elements (2D matrix of numbers).

The **output** is the **return** value of your function. Find the biggest element and return it.

Examples

Input	Output
[[20, 50, 10], [8, 33, 145]]	145

Input	Output
[[3, 5, 7, 12], [-1, 4, 33, 2], [8, 3, 0, 4]]	33

10. Diagonal Sums

A square matrix of numbers comes as an array of **strings**, each string holding numbers (space separated). Write a function that finds the sum at the main and the secondary diagonals.

The **input** comes as an **array of arrays**, containing number elements (2D matrix of numbers).

The **output** is **printed** on the console, on a single line separated by space. First print the sum at the main diagonal, then the sum at the secondary diagonal.

Example

Input	Output
[[20, 40], [10, 60]]	80 50

Input	Output
[[3, 5, 17], [-1, 7, 14], [1, -8, 89]]	99 25

11. Equal Neighbors

Write a function that finds the number of **equal neighbor** pairs inside a **matrix** of variable size and type (numbers or strings).

The **input** comes as an **array of arrays**, containing string elements (2D matrix of strings).

The **output** is **return** value of your function. Save the number of equal pairs you find and return it.

Example

Input	Output
[['2', '3', '4', '7', '0'], ['4', '0', '5', '3', '4'], ['2', '3', '5', '4', '2'], ['9', '8', '7', '5', '4']]	1

Input	Output
[['test', 'yes', 'yo', 'ho'], ['well', 'done', 'yo', '6'], ['not', 'done', 'yet', '5']]	2