

Arrays and Matrices



SoftUni Team
Technical Trainers
Software University
<http://softuni.bg>

Arrays, Array Operations, Matrices,
Multi-Dimensional Arrays

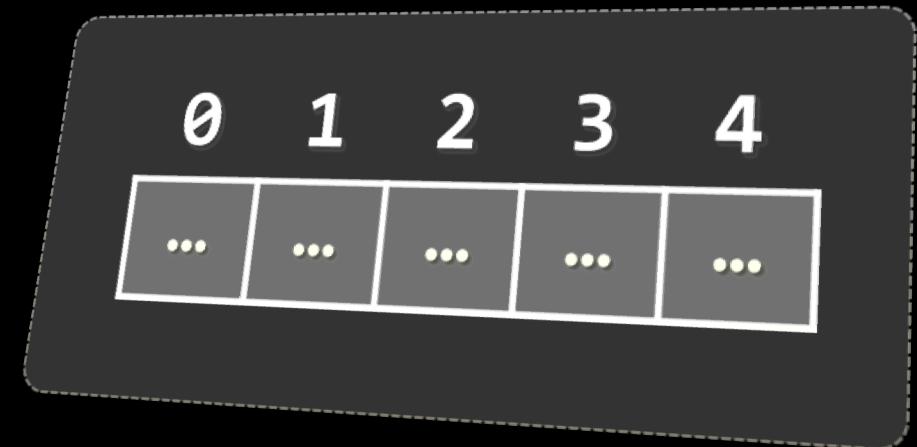


Table of Contents

1. Arrays is JavaScript

2. Array Operations

- Iteration over Arrays
- Push, Pop, Shift, Slice, Join, ...
- Sorting Arrays

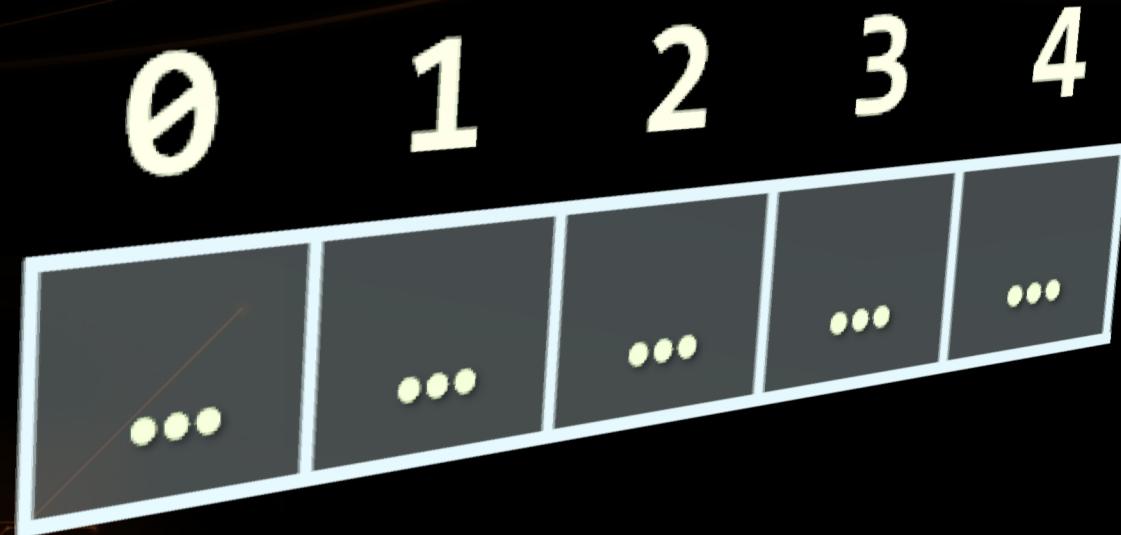
3. Matrices



Have a Question?



sli.do
#JSCORE



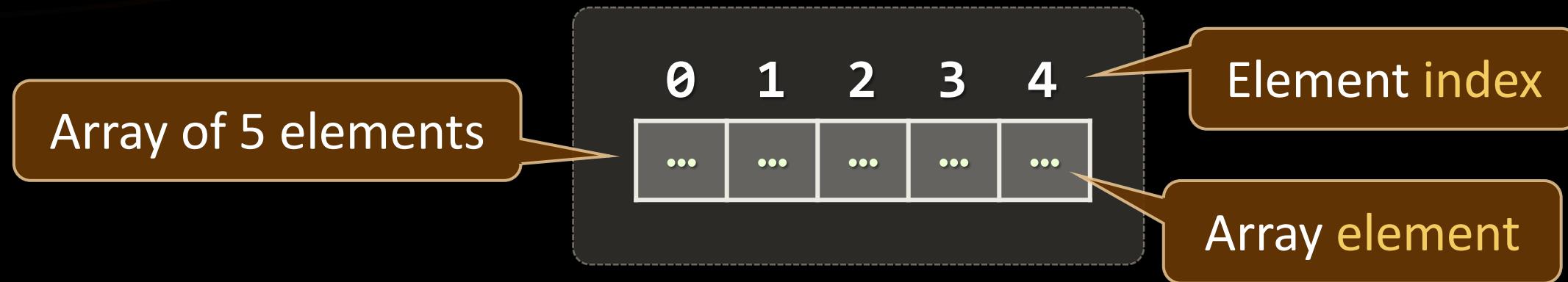
```
console.dir([10, 20, 30])
▼ Array[3] ⓘ
  0: 10
  1: 20
  2: 30
  length: 3
  ► __proto__: Array[0]
```

Arrays in JS

Working with Arrays of Elements

What are Arrays?

- In JS **arrays** are ordered sequences of elements



- Elements are numbered from **0** to **length-1**
- Arrays hold **key-value pairs**
 - **Key (0...length-1)**, **value** of any type (e.g. number / string / object)
- Arrays have **variable size** – can be resized (unlike C# / Java / C++)

Arrays – Examples

```
let arr = [10, 20, 30];
console.log(arr); // [10, 20, 30]
console.log(arr.length); // 3
console.log(arr[0]); // 10
```

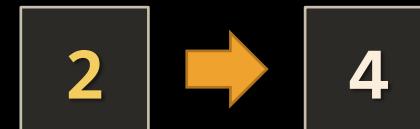
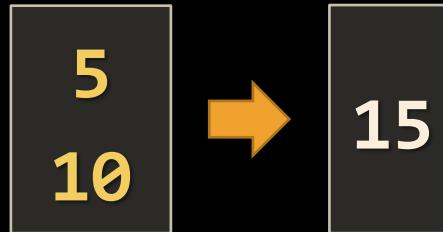
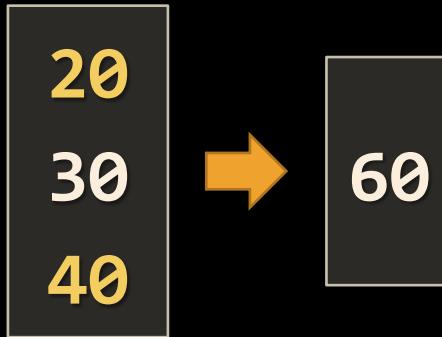
```
▼ [10, 20, 30] ⓘ
  0: 10
  1: 20
  2: 30
  length: 3
▶ __proto__: Array[0]
```

```
arr[0] = 5; // Elements can be modified
console.log(arr); // [5, 20, 30]
```

```
arr.push(500); // Elements are resizable
console.log(arr); // [5, 20, 30, 500]
```

Problem: Sum First and Last Array Elements

- You are given array of strings holding numbers
 - Calculate and print the sum of the first and the last elements



```
function sumFirstAndLast(arr) {  
    return Number(arr[0]) + Number(arr[arr.length - 1]);  
}  
  
sumFirstAndLast(['20', '30', '40']) // 60
```

Check your solution here: <https://judge.softuni.bg/Contests/311>

Processing Arrays Elements

```
let capitals = [ 'Sofia', 'Washington', 'London', 'Paris' ];
```

```
for (let capital of capitals)  
  console.log(capital);
```

for...of works like foreach

```
for (let i in capitals)  
  console.log(i + " " + capitals[i]);
```

for...in goes through array indices

```
for (let i=0; i < capitals.length; i++)  
  console.log(capitals[i]);
```

Traditional for-loop

Processing Arrays Elements (2)

```
let capitals = [ 'Sofia', 'Washington', 'London', 'Paris' ];
```

```
capitals.forEach(capital => console.log(capital));
```

```
capitals.forEach((capital, i) =>  
  console.log(i + ' -> ' + capital));
```

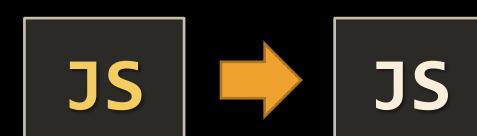
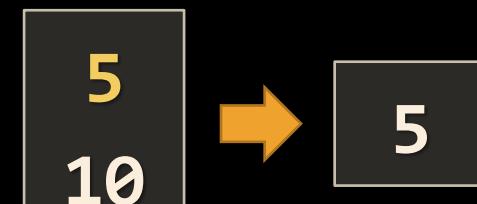
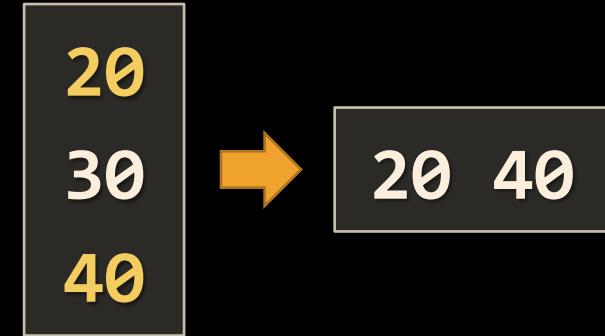
```
console.log(capitals.join(', '));
```

```
console.log(JSON.stringify(capitals));
```

Problem: Even Position Elements

- Find the elements at even positions in array, space separated

```
function evenPositions(arr) {  
    let result = [];  
    for (let i in arr)  
        if (i % 2 == 0)  
            result.push(arr[i]);  
    return result.join(' ');\n}
```



Check your solution here: <https://judge.softuni.bg/Contests/311>

Arrays of Different Types

```
// Array holding numbers
let numbers = [10, 20, 30, 40, 50];
```

```
// Array holding strings
let weekDays = ['Monday', 'Tuesday', 'Wednesday',
    'Thursday', 'Friday', 'Saturday', 'Sunday'];
```

```
// Array holding mixed data
var mixedArr =
[20, new Date(), 'hello', {x:5, y:8}];
```

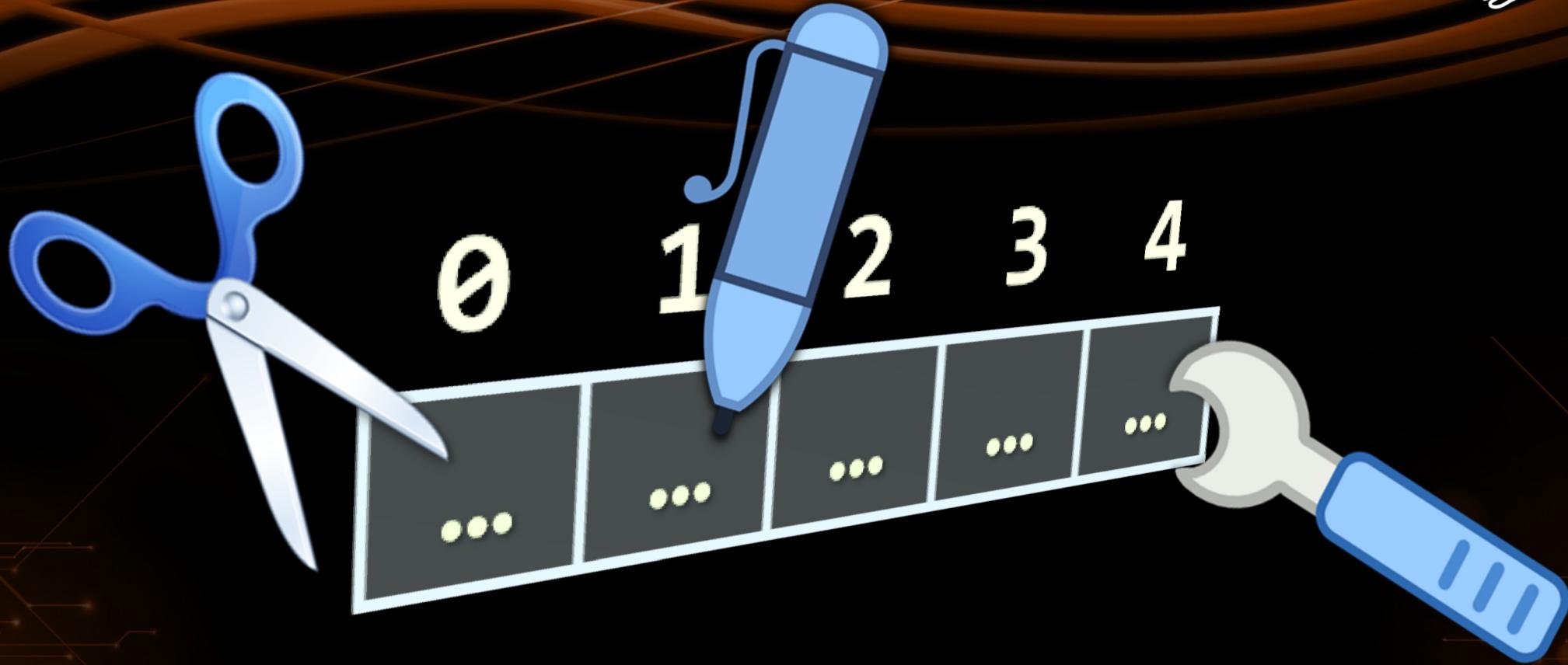
JS Arrays and Invalid Positions

```
let nums = [10, 20, 30];
nums[4] = 50; // Will resize the array
console.log(nums); // [10, 20, 30, ,50]
console.log(nums.length); // 5
console.log(nums[3]); // undefined
```

```
▼ Array[5] ⓘ
  0: 10
  1: 20
  2: 30
  4: 50
  length: 5
▶ __proto__: Array[0]
```

```
console.log(nums[-5]); // undefined
nums[-5] = -5; // Will not resize the array (invalid index)!
console.log([nums[-5], nums.length]); // [-5, 5]
```

```
console.log(nums[100]); // undefined
nums[100] = 100; // Will resize the array
console.log([nums[100], nums.length]); // [100, 101]
```



Array Operations

Push, Pop, Shift, Unshift, Slice, Join, ...

Add / Remove Elements at Both Ends

```
let nums = [10, 20, 30];
console.log(nums.join('|')); // 10|20|30
```

```
nums.push(40);
console.log(nums.join('|')); // 10|20|30|40
```

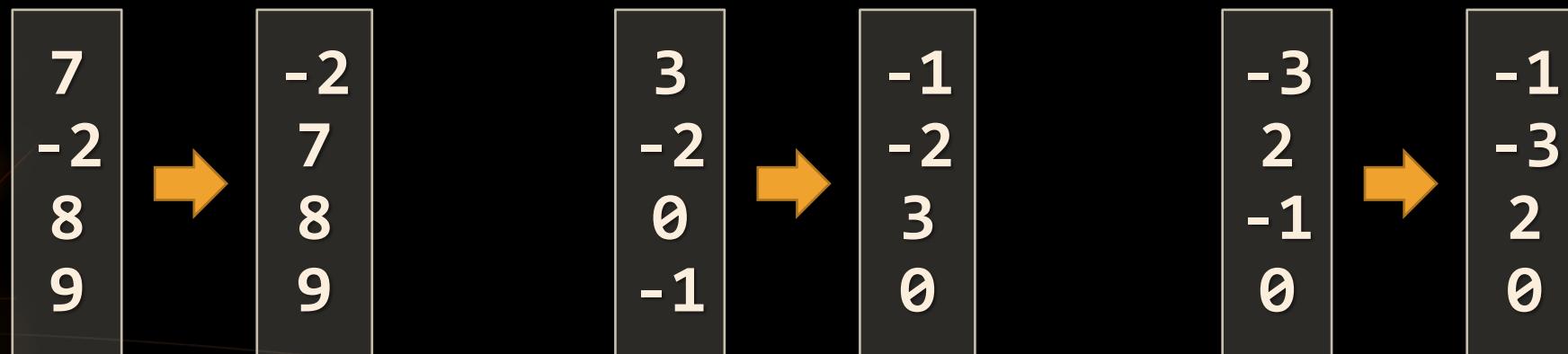
```
let tail = nums.pop();           // tail = 40
console.log(nums.join('|')); // 10|20|30
```

```
nums.unshift(0);
console.log(nums.join('|')); // 0|10|20|30
```

```
let head = nums.shift();        // head = 0
console.log(nums.join('|')); // 10|20|30
```

Problem: Negative / Positive Numbers

- You are given an array of numbers **arr**
 - Process them one by one and produce a new array **result**
 - Prepend each **negative** element at the **front** of result
 - Append each **positive** (or **0**) element at the **end** of result
 - Print the **result** array, each element at separate line



Solution: Negative / Positive Numbers

```
function negativePositiveNumbers(arr) {  
    let result = [];  
    for (num of arr)  
        if (num < 0)  
            result.unshift(num); // Insert at the start  
        else  
            result.push(num); // Append at the end  
    console.log(result.join('\n'));  
}
```

Check your solution here: <https://judge.softuni.bg/Contests/311>

Slicing Arrays

```
let nums = ['one', 'two', 'three', 'four'];
console.log(nums.join('|')); // one|two|three|four
```

```
let firstNums = nums.slice(0, 2); // start, end+1
console.log(firstNums.join('|')); // one|two
```

```
let lastNums = nums.slice(2, 4); // start, end+1
console.log(lastNums.join('|')); // three|four
```

```
let midNums = nums.slice(1, 3); // start, end+1
console.log(midNums.join('|')); // two|three
```

Splice: Cut and Insert Array Elements

```
let nums = [5, 10, 15, 20, 25, 30];
console.log(nums.join('|')); // 5|10|15|20|25|30
```

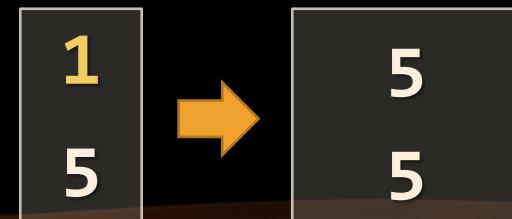
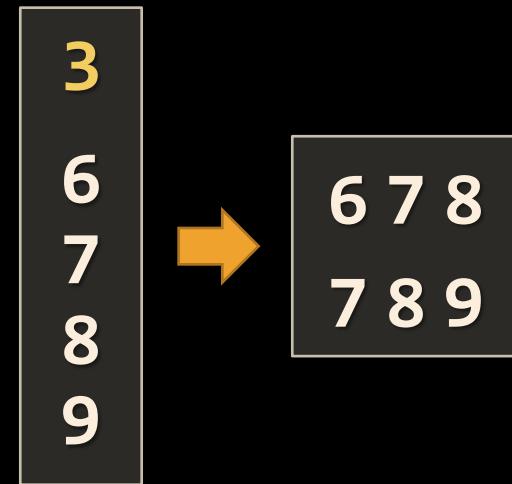
```
let mid = nums.splice(2, 3); // start, delete-count
console.log(mid.join('|')); // 15|20|25
console.log(nums.join('|')); // 5|10|30
```

```
nums = [5, 10, 15, 20, 25, 30];
nums.splice(3, 2, "twenty", "twenty-five");
console.log(nums.join('|'));
// 5|10|15|twenty|twenty-five|30
```

Problem: First and Last K Numbers

- You are given an array of numbers
 - The first element holds an integer **k**
 - Print the **first k** and the **last k** from the other elements in the array (space separated)

```
function firstLastKElements(arr) {
  let k = arr.shift();
  console.log(arr.slice(0, k).join(' '));
  console.log(arr.slice(arr.length-k,
    arr.length).join(' '));
}
```



Check your solution here: <https://judge.softuni.bg/Contests/311>

Problem: Sum Last K Numbers Sequence

- Take two integers **n** and **k**
- Generate and print the following sequence:
 - The first element is: **1**
 - All other elements = sum of the previous **k** elements
- Example: **n = 9, k = 5**
- $120 = 4 + 8 + 16 + 31 + 61$

6
3



Sequence:
1 1 2 4 7 13

8
2



Sequence:
1 1 2 3 5 8 13 21

9
5



Sequence:
1 1 2 4 8 16 31 61 120

1 1 2 | 4 8 16 31 61 | 120

+

Check your solution here: <https://judge.softuni.bg/Contests/311>

Solution: Sum Last K Numbers Sequence

```
function sumLastKNumbersSequence(n, k) {  
    let seq = [1];  
    for (let current = 1; current < n; current++) {  
        let start = Math.max(0, current - k);  
        let end = current - 1;  
        let sum = // TODO: sum the values of seq[start ... end]  
        seq[current] = sum;  
    }  
    console.log(seq.join(' '));  
}
```

Check your solution here: <https://judge.softuni.bg/Contests/311>

Filtering and Transforming Elements

```
let nums = ['one', 'two', 'three', 'four'];
console.log(nums.join('|')); // one|two|three|four
```

```
let filteredNums =
  nums.filter(x => x.startsWith('t'));
console.log(filteredNums.join('|')); // two|three
```

```
let lengths = nums.map(x => x.length);
console.log(lengths.join('|')); // 3|3|5|4
```

```
let lengths = nums.map(x => [x.length, x[0]]);
console.log(lengths.join('|')); // 3,o|3,t|5,t|4,f
```

Problem: Process Odd Numbers

- You are given an array of numbers
 - Print the odd numbers, doubled and reversed

```
function firstLastKElements(arr) {  
    let result = arr  
        .filter((num, i) => i % 2 == 1)  
        .map(x => 2*x)  
        .reverse();  
    return result.join(' ');\n}
```

10
15
20
25



50 30

3
0
10
4
7
3



6 8 0

Check your solution here: <https://judge.softuni.bg/Contests/311>



Sorting Arrays

Arranging Elements in Increasing Order

Sorting Arrays

```
let nums = [20, 40, 10, 30, 100, 5];
console.log(nums.join(' | ')); // 20|40|10|30|100|5
```

nums.sort(); // Works incorrectly on arrays of numbers !!!

```
nums.sort(); // Works incorrectly on arrays of numbers !!!
console.log(nums.join(' | ')); // 10|100|20|30|40|5
```

nums.sort((a, b) => a-b); // Compare elements as numbers

```
nums.sort((a, b) => a-b); // Compare elements as numbers
console.log(nums.join(' | ')); // 5|10|20|30|40|100
```

Problem: Smallest 2 Numbers

- You are given an array of numbers
 - Print the smallest two numbers

```
function smallestTwoNumbers(arr) {  
    arr.sort((a, b) => a-b);  
    let result = arr.slice(0, 2);  
    return result.join(' ');  
}
```

30
15
50
5

5 15

3
0
10
4
7
3

0 3

Check your solution here: <https://judge.softuni.bg/Contests/311>

	0	1	2	3
0	4	-6	3	0
1	2	1	-2	12
2	-5	17	9	5

Matrices (Tables)

Arrays Holding Arrays

Matrices in JS

- A **matrix** is a table of values

- Represented as nested arrays in JavaScript

Matrix of 4 rows

Element
`matrix[2][0]` at
 row 2, column 0

	0	1	2	3
0	4	-6	3	0
1	2	1	-2	
2	-5	17		
3	7	3	-9	12

```
var matrix = [
  [4, -6, 3, 0],
  [2, 1, -2],
  [-5, 17],
  [7, 3, -9, 12]
];
```

Matrix – Example

- Defining a matrix (array of arrays)

```
var matrix = [
  ['0,0', '0,1', '0,2'],
  ['1,0', '1,1', '1,2'],
  ['2,0', '2,1', '2,2']
];
```

- Printing a matrix

```
console.log(
  matrix.map(row => row.join(' '))
  .join('\n'));
```

Join the cells in each row by ' ', then join the rows by '\n')

```
▼ Array[3] ⓘ
  ► 0: Array[3]
    ▼ 1: Array[3]
      0: "1,0"
      1: "1,1"
      2: "1,2"
      length: 3
    ► __proto__: Array[0]
  ► 2: Array[3]
    length: 3
  ► __proto__: Array[0]
```

0,0	0,1	0,2
1,0	1,1	1,2
2,0	2,1	2,2

Problem: Biggest Element in Matrix

- You receive a 2D matrix of numbers as an array
 - Each element of the input array is an array of numbers
- Write a JS function to find the biggest number

3	5	17	12	91	5
-1	7	4	33	6	22
1	8	99	3	10	43

3	5	7	12
-1	4	33	2
8	3	0	4

20	50	10
8	33	145



99



33



145

Solution: Biggest Element in Matrix

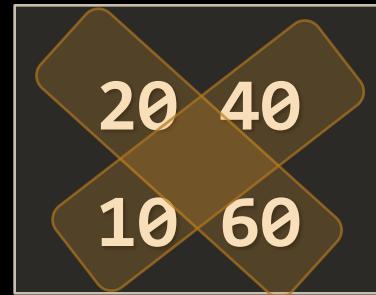
```
function biggestElement(matrix) {  
    let biggestNum = Number.NEGATIVE_INFINITY;  
    matrix.forEach(  
        r => r.forEach(  
            c => biggestNum = Math.max(biggestNum, c)));  
    return biggestNum;  
}  
  
biggestElement([[20, 50, 10], [8, 33, 145]]); // 145
```

20	50	10
8	33	145

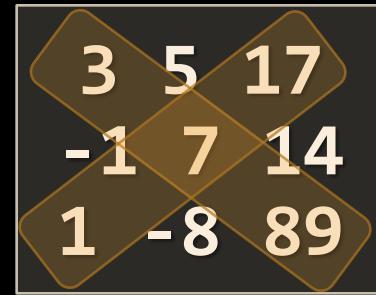
Check your solution here: <https://judge.softuni.bg/Contests/311>

Problem: Diagonal Sums

- You receive a 2D matrix of numbers as an array
 - Each element of the input array is an array of numbers
- Find sum at the main and at the secondary diagonals



80 50



99 25

Solution: Diagonal Sums

```
function diagonalSums(matrix) {  
  
    let mainSum = 0, secondarySum = 0;  
    for (let row = 0; row < matrix.length; row++) {  
        mainSum += matrix[row][row];  
        secondarySum += matrix[row][matrix.length-row-1];  
    }  
    console.log(mainSum + ' ' + secondarySum);  
}  
  
diagonalSums([[20, 40], [10, 60]]); // 80 50
```

3	5	17
-1	7	14
1	-8	89

Check your solution here: <https://judge.softuni.bg/Contests/311>

Problem: Equal Neighbors

- You are given a matrix of elements
 - Find the number of equal neighbors

2	3	4	7	0
4	0	5	3	4
2	3	5	4	2
9	8	7	5	4

1

2	2	5	7	4
4	0	5	3	4
2	5	5	4	2

5

test	yes	yo	ho
well	done	yo	6
not	done	yet	5

2

Problem: Equal Neighbors

```
function equalNeighborsCount(matrix) {  
    let neighbors = 0;  
    for (let row = 0; row < matrix.length-1; row++)  
        for (let col = 0; col < matrix[row].length; col++)  
            if (matrix[row][col] == matrix[row + 1][col])  
                neighbors++;  
    // TODO: check also the horizontal neighbors  
    return neighbors;  
}
```

2	2	5	7	4
4	0	5	3	4
2	5	5	4	2

Check your solution here: <https://judge.softuni.bg/Contests/311>



Practice: Arrays and Matrices

Live Exercises in Class (Lab)

Summary

- Arrays in JS hold sequence of elements

```
let nums = [10, 20, 30, 40];
```

- Elements are accessed by index

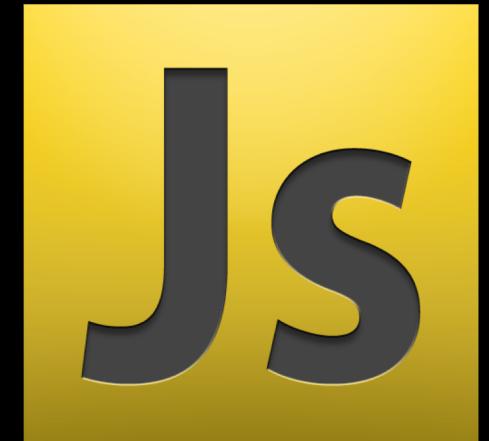
```
nums[2] = 100;
```

- Behave like lists (add / delete elements)

```
nums.push('new element');
```

- Matrices are arrays holding arrays

```
let matrix = [[10, 20], [30, 40]];
```



Arrays and Matrices



Questions?

SUPERHOSTING.BG

INDEAVR
Serving the high achievers



License

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license



Free Trainings @ Software University

- Software University Foundation – softuni.org
- Software University – High-Quality Education, Profession and Job for Software Developers
 - softuni.bg
- Software University @ Facebook
 - facebook.com/SoftwareUniversity
- Software University Forums
 - forum.softuni.bg

