# **Лабораторна робота №3**

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

1

*ДУ «Житомирська політехніка».23.121.03.00 – Лр3*

Розроб.

Білотіл Я. О.

Перевір.

Дмитренко І. А.

*Т. контр.*

Н. контр.

Затверд.

Звіт

з лабораторної роботи

Літ.

Аркушів

*13*

*ФІКТ Гр. IПЗк-23-1*

# **Тема**: Робота з одновимірними та багатовимірними масивами.

**Завдання 0. Ознайомитись з теоретичними відомостями за посиланням:**

[Масиви](https://uk.javascript.info/array)

[Методи масивів](https://uk.javascript.info/array-methods)

**Завдання 1.**

**1.1.** Розмістіть на сторінці **div**-блок з ідентифікатором **“paragraphs”**. Розмістіть у ньому 5 абзаців з довільним текстом. Сформуйте масив, який міститиме 5 елементів - кількість символів, які входять до відповідного абзацу тексту.

Для пошуку абзаців використовуйте метод **querySelectorAll** та відповідний CSS-селектор. Для доступу до тексту, який входить в абзац використовуйте властивість **innerText** для DOM-елемента. Виведіть у консоль отриманий масив.

Код програми:

index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Завдання 1.1</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<div id="paragraphs">

<p>Сонце спроможне зігріти землю та подарувати нам світло і тепло.</p>

<p>У лісі шепочуть вітри, а дерева танцюють під їхній мелодії.</p>

<p>Музика може вражати душу, викликати емоції та переносити нас в інші світи.</p>

<p>Книги — це вікна в інші реальності, де кожна сторінка розкриває нову пригоду.</p>

<p>Усмішка — це найпрекрасніший спосіб поділитися теплом та радістю з оточуючими.</p>

</div>

<script>

const paragraphsDiv = document.getElementById("paragraphs");

const paragraphs = paragraphsDiv.querySelectorAll("p");

const characterCounts = [];

paragraphs.forEach(paragraph => {

const text = paragraph.innerText;

const characterCount = text.length;

characterCounts.push(characterCount);

});

console.log(characterCounts);

</script>

</body>

</html>

style.css

#paragraphs {

width: 60%;

margin: 0 auto;

padding: 20px;

border: 1px solid #ccc;

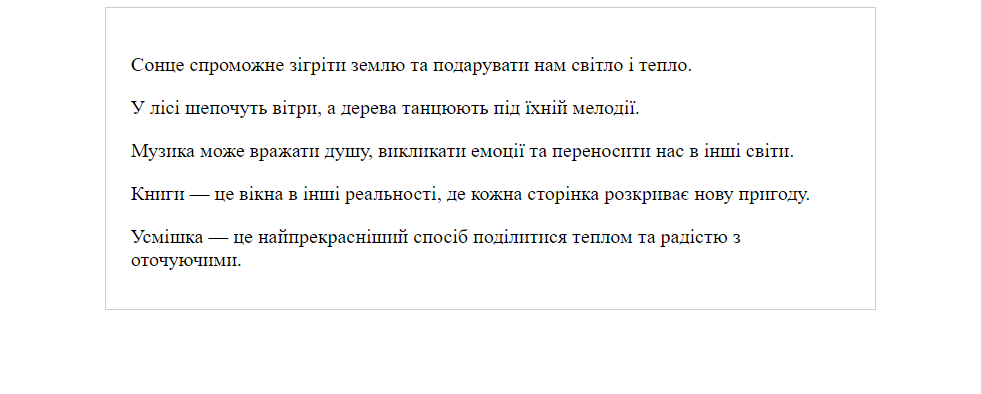
}

p {

margin-bottom: 10px;

}

Результат роботи програми:



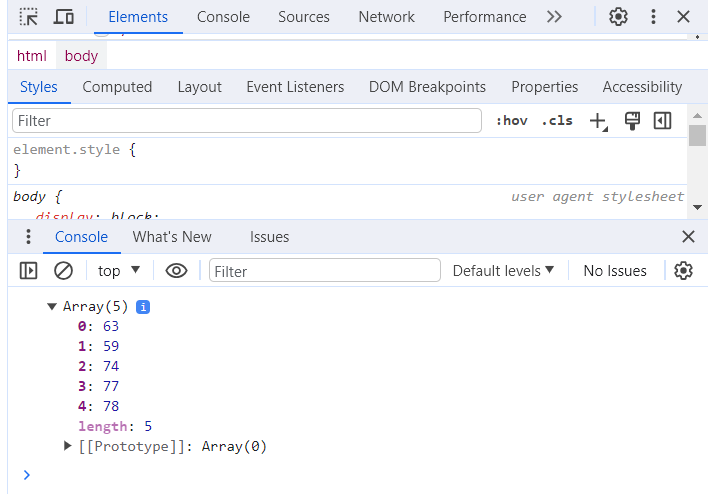


Рис.1 – Завдання 1.1

**1.2.**  Дано масив цілих чисел:

let arr = [20, 17, 4, -4, 10, -9, 13, 4, 12, 22, 13, 19, 1, 3];

- Для заданого масиву визначте та виведіть у консоль:

* найменше та найбільше значення;
* кількість парних та непарних чисел;
* кількість чисел, що містять дві цифри;
* кількість додатних та кількість від’ємних чисел.

- Відсортуйте масив **arr** за допомогою методу Sort у порядку зростання значень та виведіть у консоль результат.

- Створіть масив з оберненим порядок елементів та виведіть його у консоль.

Код програми:

index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="main.js"></script>

</head>

<body>

</body>

</html>

main.js

let arr = [20, 17, 4, -4, 10, -9, 13, 4, 12, 22, 13, 19, 1, 3];

const min = Math.min(...arr);

const max = Math.max(...arr);

console.log(`Мінімальне значення: ${min}`);

console.log(`Максимальне значення: ${max}`);

const evenCount = arr.filter(num => num % 2 === 0).length;

const oddCount = arr.length - evenCount;

console.log(`Кількість парних чисел: ${evenCount}`);

console.log(`Кількість непарних чисел: ${oddCount}`);

const twoDigitCount = arr.filter(num => Math.abs(num) >= 10 && Math.abs(num) <= 99).length;

console.log(`Кількість чисел, що містять дві цифри: ${twoDigitCount}`);

const positiveCount = arr.filter(num => num > 0).length;

const negativeCount = arr.filter(num => num < 0).length;

console.log(`Кількість додатних чисел: ${positiveCount}`);

console.log(`Кількість від’ємних чисел: ${negativeCount}`);

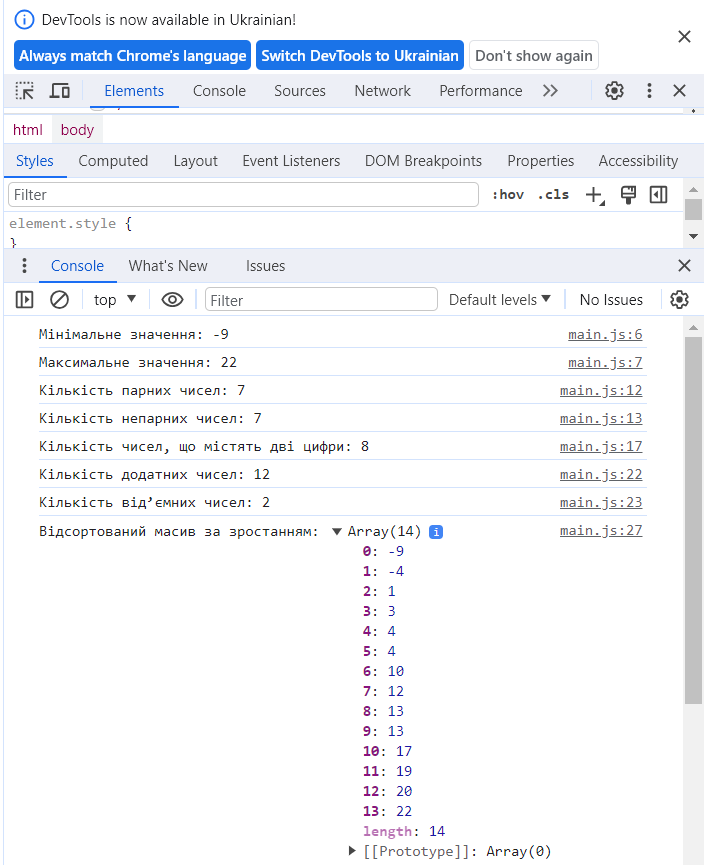
arr.sort((a, b) => a - b);

console.log('Відсортований масив за зростанням:', arr);

const reversedArr = arr.slice().reverse();

console.log('Масив з оберненим порядком елементів:', reversedArr);

Результат роботи програми:



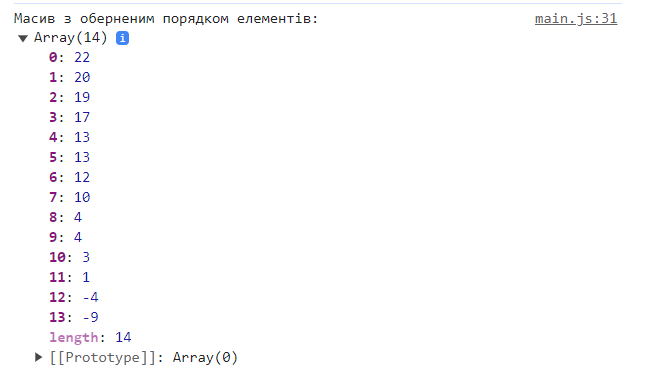


Рис.2 – Завдання 1.2

**Завдання 2.**

* Розмістіть на сторінці 3 div-блоки з класами “**table”**. У кожному з них розмістіть табличку. У першому блоці - табличку розміром 3х3, другому - 4х4, третьому - 5х5.
* В CSS-файлі розмістіть стиль для класу “**selected**”, який передбачає жовтий фон.
* За допомогою **querySelectorAll** знайдіть **div**-блоки з класом “**table**”. Для кожного знайденого блоку у циклі знайдіть всі вкладені у нього теги **td**. Для кожного другого тега **td** додайте клас “**selected**”.

Код програми:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="styles.css">

<title>Table Blocks</title>

<link rel="stylesheet" href="style.css">

<script src="script.js"></script>

</head>

<body>

<div class="table">

<table>

<tr>

<td>1</td>

<td>2</td>

<td>3</td>

</tr>

<tr>

<td>4</td>

<td>5</td>

<td>6</td>

</tr>

<tr>

<td>7</td>

<td>8</td>

<td>9</td>

</tr>

</table>

</div>

<div class="table">

<table>

<tr>

<td>1</td>

<td>2</td>

<td>3</td>

<td>4</td>

</tr>

<tr>

<td>5</td>

<td>6</td>

<td>7</td>

<td>8</td>

</tr>

<tr>

<td>9</td>

<td>10</td>

<td>11</td>

<td>12</td>

</tr>

<tr>

<td>13</td>

<td>14</td>

<td>15</td>

<td>16</td>

</tr>

</table>

</div>

<div class="table">

<table>

<tr>

<td>1</td>

<td>2</td>

<td>3</td>

<td>4</td>

<td>5</td>

</tr>

<tr>

<td>6</td>

<td>7</td>

<td>8</td>

<td>9</td>

<td>10</td>

</tr>

<tr>

<td>11</td>

<td>12</td>

<td>13</td>

<td>14</td>

<td>15</td>

</tr>

<tr>

<td>16</td>

<td>17</td>

<td>18</td>

<td>19</td>

<td>20</td>

</tr>

<tr>

<td>21</td>

<td>22</td>

<td>23</td>

<td>24</td>

<td>25</td>

</tr>

</table>

</div>

</body>

</html>

style.css

.table {

margin-bottom: 20px;

}

.table table {

border-collapse: collapse;

}

.table table td {

border: 1px solid black;

padding: 10px;

}

.selected {

background-color: yellow;

}

script.js

document.addEventListener("DOMContentLoaded", function () {

const tableDivs = document.querySelectorAll('.table');

tableDivs.forEach(tableDiv => {

const tds = tableDiv.querySelectorAll('td');

tds.forEach((td, index) => {

if (index % 2 !== 0) {

td.classList.add('selected');

}

});

});

});

Результат роботи програми:

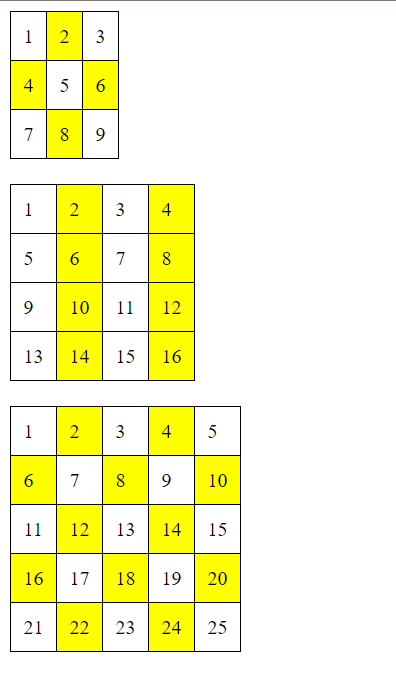


Рис.3 – Завдання 2

**Завдання 3.**

Створіть матрицю розміру N на M, яка містить цілі числа (наприклад, 7х5 та числами на власний розсуд). Знайдіть та виведіть у консоль:

* кількість додатних елементів;
* кількість рядків, які не містять жодного нульового елемента;
* кількість стовпців, які містять хоча б один нульовий елемент;
* номер рядка, в якому знаходиться найдовша серія однакових елементів;
* добуток елементів в тих рядках, які не містять від’ємних елементів;
* суму елементів в тих стовпцях, які не містять від’ємних елементів;
* суму елементів в тих стовпцях, які містять хоча б один від’ємний елемент;
* транспоновану матрицю;

Код програми:

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="script.js"></script>

</head>

<body>

</body>

</html>

script.js

const N = 7;

const M = 5;

const matrix = Array.from({ length: N }, () => Array.from({ length: M }, () => Math.floor(Math.random() \* 20) - 10));

console.log("Початкова матриця:");

console.log(matrix);

console.log();

const positiveCount = matrix.flat().filter(element => element > 0).length;

console.log("Кількість додатних елементів:", positiveCount);

console.log();

const nonZeroRowsCount = matrix.filter(row => !row.includes(0)).length;

console.log("Кількість рядків без нульових елементів:", nonZeroRowsCount);

console.log();

const zeroColumnsCount = matrix[0].map((\_, colIndex) => matrix.some(row => row[colIndex] === 0)).filter(Boolean).length;

console.log("Кількість стовпців з нульовим елементом:", zeroColumnsCount);

console.log();

const maxConsecutiveRow = matrix.reduce((maxIndex, row, currentIndex) => {

const consecutiveLength = row.reduce((maxCount, \_, index, array) => {

if (index < array.length - 1 && array[index] === array[index + 1]) {

return maxCount + 1;

}

return maxCount;

}, 1);

return consecutiveLength > maxIndex.maxCount ? { maxCount: consecutiveLength, index: currentIndex } : maxIndex;

}, { maxCount: 0, index: 0 }).index;

console.log("Номер рядка з найдовшою серією однакових елементів:", maxConsecutiveRow);

console.log();

const positiveRowsProduct = matrix.map(row => row.every(element => element >= 0) ? row.reduce((acc, val) => acc \* val, 1) : 0);

console.log("Добуток елементів в рядках без від’ємних елементів:", positiveRowsProduct);

console.log();

const positiveColumnsSum = matrix[0].map((\_, colIndex) => matrix.map(row => row[colIndex]).every(element => element >= 0) ? matrix.reduce((acc, row) => acc + row[colIndex], 0) : 0);

console.log("Сума елементів в стовпцях без від’ємних елементів:", positiveColumnsSum);

console.log();

const negativeColumnsSum = matrix[0].map((\_, colIndex) => matrix.map(row => row[colIndex]).some(element => element < 0) ? matrix.reduce((acc, row) => acc + row[colIndex], 0) : 0);

console.log("Сума елементів в стовпців з від’ємним елементом:", negativeColumnsSum);

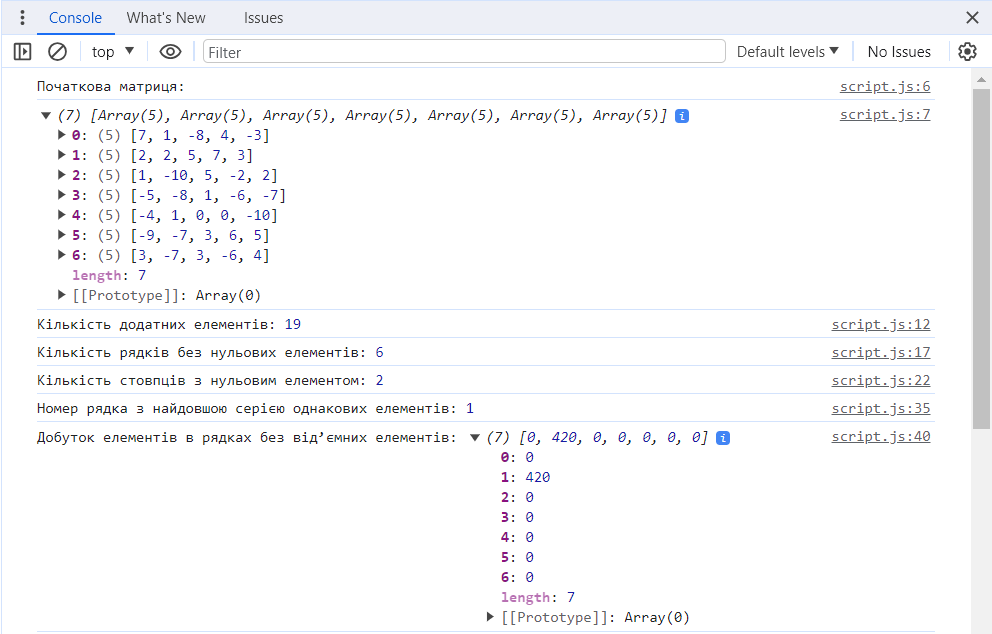
console.log();

const transposedMatrix = matrix[0].map((\_, colIndex) => matrix.map(row => row[colIndex]));

console.log("Транспонована матриця:");

console.log(transposedMatrix);

Результат роботи програми:



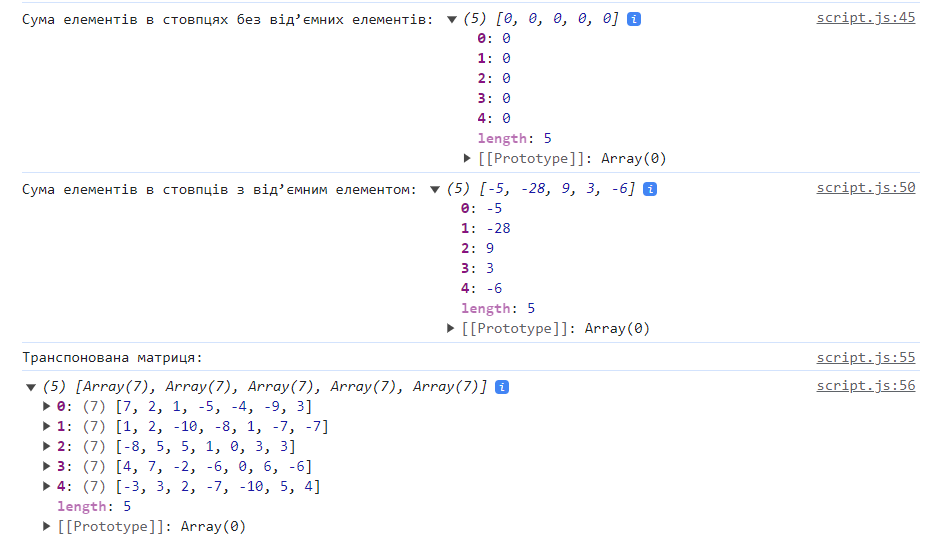


Рис.4 – Завдання 3