**Лабораторна робота №14**

**Тема: Методи масивів**

***Мета роботи:*** навчитися працювати з методами масивів та методами роботи з ними

**Завдання на лабораторну роботу**

**Завдання 0:** Ретельно вивчити теорію за посиланнями:

Регулярні вирази:

<https://uk.javascript.info/array-methods>

**Завдання 1:**

**[Переведіть текст виду border-left-width в borderLeftWidth](https://uk.javascript.info/array-methods" \l "perevedit-tekst-vidu-border-left-width-v-borderleftwidth)**

важливість: 5

Напишіть функцію camelize(str), яка перетворює такі рядки “my-short-string” в “myShortString”.

Тобто дефіси видаляються, а всі слова після них починаються з великої літери.

Приклади:

camelize("background-color") == 'backgroundColor';

camelize("list-style-image") == 'listStyleImage';

camelize("-webkit-transition") == 'WebkitTransition';

P.S. Підказка: використовуйте split, щоб розбити рядок на масив символів, потім переробіть все як потрібно та методом join зʼєднайте елементи в рядок.

**[Фільтрація за діапазоном](https://uk.javascript.info/array-methods" \l "filtraciya-za-diapazonom)**

важливість: 4

Напишіть функцію filterRange(arr, a, b), яка приймає масив arr, шукає в ньому елементи більші-рівні a та менші-рівні b і віддає масив цих елементів.

Функція повинна повертати новий масив і не змінювати вихідний.

Наприклад:

let arr = [5, 3, 8, 1];

let filtered = filterRange(arr, 1, 4);

alert( filtered ); // 3,1 (відфільтровані значення)

alert( arr ); // 5,3,8,1 (не змінюється)

**[Фільтрація за діапазоном "на місці"](https://uk.javascript.info/array-methods" \l "filtraciya-za-diapazonom-na-misci)**

важливість: 4

Напишіть функцію filterRangeInPlace(arr, a, b), яка приймає масив arr і видаляє з нього всі значення крім тих, які знаходяться між a і b. Тобто, перевірка має вигляд a ≤ arr[i] ≤ b.

Функція повинна змінювати поточний масив і нічого не повертати.

Наприклад:

let arr = [5, 3, 8, 1];

filterRangeInPlace(arr, 1, 4); // видаляє всі числа крім тих, що в діапазоні від 1 до 4

alert( arr ); // [3, 1]

**[Сортувати за спаданням](https://uk.javascript.info/array-methods" \l "sortuvati-za-spadannyam)**

важливість: 4

let arr = [5, 2, 1, -10, 8];

// ... ваш код для сортування за спаданням

alert( arr ); // 8, 5, 2, 1, -10

**[Скопіювати і впорядкувати масив](https://uk.javascript.info/array-methods" \l "skopiyuvati-i-vporyadkuvati-masiv)**

важливість: 5

У нас є масив рядків arr. Потрібно отримати відсортовану копію та залишити arr незміненим.

Створіть функцію copySorted(arr), яка буде повертати таку копію.

let arr = ["HTML", "JavaScript", "CSS"];

let sorted = copySorted(arr);

alert( sorted ); // CSS, HTML, JavaScript

alert( arr ); // HTML, JavaScript, CSS (без змін)

**[Створити розширюваний калькулятор](https://uk.javascript.info/array-methods" \l "stvoriti-rozshiryuvanii-kalkulyator)**

важливість: 5

Створіть функцію-конструктор Calculator, яка створює «розширюваний» обʼєкт калькулятора.

Завдання складається з двох частин.

1. По-перше, реалізуйте метод calculate(str), який приймає рядок типу "1 + 2" в форматі «ЧИСЛО оператор ЧИСЛО» (розділені пробілами) і повертає результат. Метод повинен розуміти плюс + і мінус -.

Приклад використання:

let calc = new Calculator;

alert( calc.calculate("3 + 7") ); // 10

1. Потім додайте метод addMethod(name, func), який додає в калькулятор нові операції. Він приймає оператор name і функцію з двома аргументами func(a, b), яка описує його.

Наприклад, давайте додамо множення \*, ділення / і зведення в ступінь \*\*:

let powerCalc = new Calculator;

powerCalc.addMethod("\*", (a, b) => a \* b);

powerCalc.addMethod("/", (a, b) => a / b);

powerCalc.addMethod("\*\*", (a, b) => a \*\* b);

let result = powerCalc.calculate("2 \*\* 3");

alert( result ); // 8

* Для цього завдання не потрібні дужки або складні вирази.
* Числа і оператор розділені рівно одним пропуском.
* Не зайвим буде додати обробку помилок.

**[Трансформувати в масив імен](https://uk.javascript.info/array-methods" \l "transformuvati-v-masiv-imen)**

важливість: 5

У вас є масив об’єктів user, і в кожному з них є user.name. Напишіть код, який перетворює їх в масив імен.

Наприклад:

let ivan = { name: "Іван", age: 25 };

let petro = { name: "Петро", age: 30 };

let mariya = { name: "Марія", age: 28 };

let users = [ ivan, petro, mariya ];

let names = /\* ... ваш код \*/

alert( names ); // Іван, Петро, Марія

**[Трансформувати в обʼєкти](https://uk.javascript.info/array-methods" \l "transformuvati-v-ob-yekti)**

важливість: 5

У вас є масив обʼєктів user, і у кожного з обʼєктів є name, surname та id.

Напишіть код, який створить ще один масив обʼєктів з параметрами id й fullName, де fullName – складається з name та surname.

Наприклад:

let ivan = { name: "Іван", surname: "Іванко", id: 1 };

let petro = { name: "Петро", surname: "Петренко", id: 2 };

let mariya = { name: "Марія", surname: "Мрійко", id: 3 };

let users = [ ivan, petro, mariya ];

let usersMapped = /\* ... ваш код ... \*/

/\*

usersMapped = [

{ fullName: "Іван Іванко", id: 1 },

{ fullName: "Петро Петренко", id: 2 },

{ fullName: "Марія Мрійко", id: 3 }

]

\*/

alert( usersMapped[0].id ) // 1

alert( usersMapped[0].fullName ) // Іван Іванко

Отже, насправді вам потрібно трансформувати один масив обʼєктів в інший. Спробуйте використовувати =>. Це невелика хитрість.

**[Відсортувати користувачів за віком](https://uk.javascript.info/array-methods" \l "vidsortuvati-koristuvachiv-za-vikom)**

важливість: 5

Напишіть функцію sortByAge(users), яка приймає масив обʼєктів з властивістю age і сортує їх по ньому.

Наприклад:

let ivan = { name: "Іван", age: 25 };

let petro = { name: "Петро", age: 30 };

let mariya = { name: "Марія", age: 28 };

let arr = [ petro, ivan, mariya ];

sortByAge(arr);

// now: [ivan, mariya, petro]

alert(arr[0].name); // Іван

alert(arr[1].name); // Марія

alert(arr[2].name); // Петро

**[Перемішайте масив](https://uk.javascript.info/array-methods" \l "peremishaite-masiv)**

важливість: 3

Напишіть функцію shuffle(array), яка перемішує (випадковим чином переставляє) елементи масиву.

Багаторазові прогони через shuffle можуть привести до різних послідовностей елементів. Наприклад:

let arr = [1, 2, 3];

shuffle(arr);

// arr = [3, 2, 1]

shuffle(arr);

// arr = [2, 1, 3]

shuffle(arr);

// arr = [3, 1, 2]

// ...

Всі послідовності елементів повинні мати однакову ймовірність. Наприклад, [1,2,3] може бути перемішана як [1,2,3] або [1,3,2], або [3,1,2] тощо, з однаковою ймовірністю кожного випадку.

**[Вирахувати середній вік](https://uk.javascript.info/array-methods" \l "virakhuvati-serednii-vik)**

важливість: 4

Напишіть функцію getAverageAge(users), яка приймає масив об’єктів з властивістю age та повертає середній вік.

Формула обчислення середнього арифметичного значення: (age1 + age2 + ... + ageN) / N.

Наприклад:

let john = { name: "John", age: 25 };

let pete = { name: "Pete", age: 30 };

let mary = { name: "Mary", age: 29 };

let arr = [ john, pete, mary ];

alert( getAverageAge(arr) ); // (25 + 30 + 29) / 3 = 28

**[Залишити унікальні елементи масиву](https://uk.javascript.info/array-methods" \l "zalishiti-unikalni-elementi-masivu)**

важливість: 4

Нехай arr – масив рядків.

Напишіть функцію unique(arr), яка повертає масив, що містить тільки унікальні елементи arr.

Наприклад:

function unique(arr) {

/\* ваш код \*/

}

let strings = ["Привіт", "Світ", "Привіт", "Світ",

"Привіт", "Привіт", "Світ", "Світ", ":-O"

];

alert( unique(strings) ); // Привіт, Світ, :-O

**[Створення об’єкта з ключем з масиву](https://uk.javascript.info/array-methods" \l "stvorennya-obyekta-z-klyuchem-z-masivu)**

важливість: 4

Припустимо, ми отримали масив користувачів у вигляді {id:..., name:..., age:...}.

Створіть функцію groupById(arr), яка створює з масиву об’єкт із ключом id та елементами масиву як значеннями.

Наприклад:

let users = [

{id: 'іван', name: "Іван Іванко", age: 20},

{id: 'ганна', name: "Ганна Іванко", age: 24},

{id: 'петро', name: "Петро Петренко", age: 31},

];

let usersById = groupById(users);

/\*

// після виклику функції ви повинні отримати:

usersById = {

іван: {id: 'іван', name: "Іван Іванко", age: 20},

ганна: {id: 'ганна', name: "Ганна Іванко", age: 24},

петро: {id: 'петро', name: "Петро Петренко", age: 31},

}

\*/

Така функція дійсно зручна при роботі з даними сервера.

У цьому завданні ми вважаємо, що id унікальний. Не може бути двох елементів масиву з однаковими id.

Будь ласка, використовуйте метод масиву .reduce у рішенні.

*Закомітити виконані завдання. Надати доступ до репозиторію викладачам*