ЛАБОРАТОРНА РОБОТА №2

JavaScript. Data types, typeof, type casting, arrays, date, object

Мета: навчитися працювати з масивами та датами у JavaScript, закріпити навички перевірки типів і валідації введених даних.

Варіант 10

Дано дві дати з часом. Обчислити різницю у годинах і хвилинах.

```
const readline = require("readline");

const rl = readline.createInterface({
    input: process.stdin,
    output: process.stdout
});



console.log("Обчислення різниці між двома датами у годинах і хвилинах.");
console.log("Введи дати у форматі: YYYY-MM-DD HH:MM\n");



function isValidDate(dateStr) {
    if (typeof dateStr !== "string") return false;
    const d = new Date(dateStr);
    return !isNaN(d.getTime());
}

function calculateTimeDifference(date1, date2) {
    let d1 = new Date(date1);
    let d2 = new Date(date2);
    let diffMs = Math.abs(d2 - d1);

    let diffHours = Math.floor(diffMs / (1000 * 60 * 60));
    let diffMinutes = Math.floor((diffMs % (1000 * 60 * 60)) / (1000 * 60));

    console.log(`\nДата 1: ${d1}`);
    console.log(`Дата 2: ${d2}`);
    console.log(`Різниця: ${diffHours} годин ${diffMinutes} хвилин`);
}



rl.question("Введіть першу дату: ", (date1) => {
    if (!isValidDate(date1)) {
        console.log(" Помилка: некоректна дата! Формат повинен бути YYYY-MM-DD
HH:MM");
```

```
        rl.close();
        return;
    }

    rl.question("Введіть другу дату: ", (date2) => {
        if (!isValidDate(date2)) {
            console.log(" Помилка: некоректна дата! Формат повинен бути YYYY-
MM-DD HH:MM");
            rl.close();
            return;
        }

        calculateTimeDifference(date1, date2);
        rl.close();
    });
});
```

PS D:\lab2_js> node laba2.js

Обчислення різниці між двома датами у годинах і хвилинах.

Введи дати у форматі: YYYY-MM-DD HH:MM

Введіть першу дату: 2025-10-05 08:30

Введіть другу дату: 2025-10-05 15:45

Дата 1: Sun Oct 05 2025 08:30:00 GMT+0300 (за східноєвропейським літнім часом)

Дата 2: Sun Oct 05 2025 15:45:00 GMT+0300 (за східноєвропейським літнім часом)

Різниця: 7 годин 15 хвилин

PS D:\lab2_js>

Згенерувати масив із 10 випадкових чисел. Відсортувати масив у порядку зростання.

```
function generateAndSortArray() {


    let arr = [];


    for (let i = 0; i < 10; i++) {
        let num = Math.floor(Math.random() * 100);
```

```
        if (isNaN(num)) {
            console.log(` Помилка: згенероване значення №${i + 1} не є
числом`);
            return;
        }

        arr.push(num);
    }


    for (let i = 0; i < arr.length; i++) {
        if (typeof arr[i] !== "number") {
            console.log(` Помилка: елемент ${i + 1} не є числом`);
            return;
        }
    }

    console.log("Згенерований масив:", arr);


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

    console.log("Відсортований масив:", sorted);
}


generateAndSortArray();
```

PS D:\lab2_js> node laba2.js

Згенерований масив: [

  39, 41, 53, 96, 25,

  57, 37, 51, 45, 10

]

Відсортований масив: [

  10, 25, 37, 39, 41,

  45, 51, 53, 57, 96

]

PS D:\lab2_js>

Контрольні питання:

Які типи даних існують у JavaScript і які з них належать до примітивних?

Примітивні — number, bigint, string, boolean, undefined, null, symbol.

Об’єктні — object, function.

Чим відрізняється значення null від undefined у плані використання та перевірки?

null — задається явно як “порожнє значення”.

undefined — означає, що змінна створена, але їй нічого не присвоїли.

Який результат дасть вираз: typeof NaN? Чому саме такий?

typeof NaN - "number". NaN — це спеціальне числове значення, що означає “не число”.

Яке значення поверне Number(" ") і чому?

Number(" ") - 0, бо порожній рядок після приведення сприймається як 0.

Який результат дає Boolean("0") і чим він відрізняється від Boolean(0)?

Boolean("0") - true, бо непорожній рядок завжди true. Boolean(0) → false, бо саме число 0 є “хибним” значенням.

6. У чому полягає різниця між масивом і об’єктом у JavaScript, якщо обидва є об’єктами?

Масив — це впорядкована колекція елементів із числовими індексами.
Об’єкт — невпорядкана колекція пар “ключ–значення”.

7. Який результат виконання коду? Поясніть, у чому різниця між slice і splice. let arr = [1, 2, 3]; console.log(arr.slice(1, 2)); console.log(arr.splice(1, 2)); console.log(arr);

console.log(arr.slice(1, 2)); // [2] — копіює

console.log(arr.splice(1, 2)); // [2,3] — видаляє з оригіналу

8. Як за допомогою Math.random() отримати випадкове ціле число в діапазоні від -50 до 50 включно?

Math.floor(Math.random() * 101) - 50;

9. Який результат обчислень наступного коду? Чому саме так? let d1 = new Date("2025-01-01"); let d2 = new Date("2025-01-02"); console.log((d2 - d1) / (1000 * 60 * 60 * 24));

Результат обчислень буде 1.

Коли ми віднімаємо два об'єкти Date в JavaScript, вони автоматично перетворюються на свої числові значення (timestamp у мілісекундах від 1 січня 1970 року)

d1 = new Date("2025-01-01") → 1735689600000 мс

d2 = new Date("2025-01-02") → 1735776000000 мс

d2 - d1 = 1735776000000 - 1735689600000 = 86400000 мс

/(1000 * 60 * 60 * 24):

1000 — мілісекунди в секунді

60 — секунди в хвилині

60 — хвилини в годині

24 — години в добі

Разом: 1000 × 60 × 60 × 24 = 86400000 — це кількість мілісекунд в одній добі

Фінальний результат:

86400000 / 86400000 = 1

10.Який результат дасть наступний код? Чому змінилось значення в обох змінних? let obj = { a: 1 }; let copy = obj; copy.a = 5; console.log(obj.a);

Результат коду буде 5.

Присвоєння за посиланням:

let obj = { a: 1 };
let copy = obj;

Коли ми присвоюємо об'єкт іншій змінній, ми копіюємо посилання на той самий об'єкт в пам'яті, а не створюємо новий незалежний об'єкт.

Обидві змінні вказують на один об'єкт:

text

obj → { a: 1 } ← copy

Зміна через будь-яку змінну:

copy.a = 5;

Оскільки обидві змінні obj і copy посилаються на той самий об'єкт, зміна властивості через одну змінну відображається при зверненні через іншу.

Результат:

```
console.log(obj.a); // 5
console.log(copy.a); // 5
```

Як уникнути цієї поведінки:
Для створення справжньої копії об'єкта можна використовувати:

```
let copy = {...obj}; // spread operator
// або
let copy = Object.assign({}, obj);
// або
let copy = JSON.parse(JSON.stringify(obj));
```

Це демонструє фундаментальну відмінність між примітивними типами (які копіюються за значенням) та об'єктами (які копіюються за посиланням) в JavaScript.