Державний університет інтелектуальних технологій і зв’язку

Факультет: Інформаційних технологій та кібербезпеки

Кафедра: Комп’ютерних наук

**Протокол лабораторної роботи №2**

**з дисципліни**

**«Веб-технології»**

Виконав студент групи ІПЗ-3.02

Вайцеховський Олександр

Перевірив ст. викл. каф. КН

Северин М.В.

Одеса - 2025

**Лабораторної роботи №2**

**Тема:** Функції в JavaScript.

**Мета:** набуття практичного досвіду створенні функцій в JavaScript.

**Завдання**

**1.** Напишіть користувацьку функцію min(a, b), яка повертає менше з двох чисел a та b. При рішенні задачі використати умовний тернарний оператор та function declaration.

function min(a, b) {

    if(a > b) {

        return a;

    } else {

        return b;

    }

}

let firstListener  = document

  .querySelector("body")

  .addEventListener("minDataCollectedDeclaration", (cache) => {

    alert("Declaration result: a is: " + cache.detail.a + ", b is: " + cache.detail.b + ", min is: " + this.min(cache.detail.a, cache.detail.b));

});

Шмат коду 1 — Функція 1

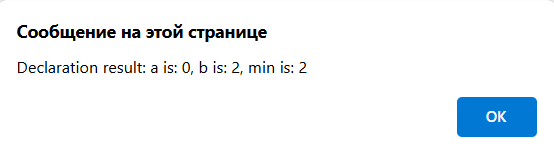


Рисунок 1 — Результат виконання

2. Зробіть копію рішення завдання 2 та замініть function declaration на function expression.

let secondListener = document

  .querySelector("body")

  .addEventListener("minDataCollectedExpression", (cache) => {

    let expressionValue = minExpression(cache.detail.a, cache.detail.b);

    alert("Declaration result: a is: " + cache.detail.a + ", b is: " + cache.detail.b + ", min is: " + expressionValue);

});

let minExpression = function(a, b) {

    if(a > b) {

        return a;

    } else {

        return b;

    }

};

Шмат коду 2 — Функція 2

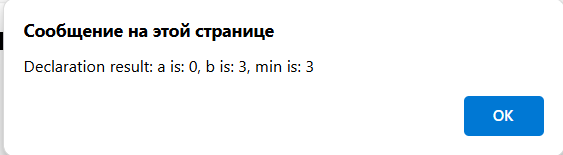


Рисунок 2 — Результат функції

3. Зробіть копію рішення завдання 3 та замініть function expression на стрілочну функцію.

let thirdListener  = document

  .querySelector("body")

  .addEventListener("minDataCollectedLambda", (cache) => {

    console.log(cache.detail);

    let min = (a, b) => a < b ? a : b;

    alert("Lambda result: a is: " + cache.detail.a + ", b is: " + cache.detail.b + ", min is: " + min(cache.detail.a, cache.detail.b));

});

Шмат коду 3 — Функція 3

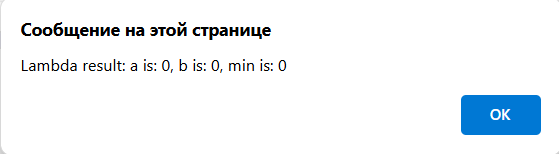


Рисунок 3 — Результат функції

4. Створіть function declaration яка може приймати три параметра: прізвище, ім’я, по батькові і формує з отриманих даних об’єкт з відповідними полями та значеннями. Врахуйте, що не всі мають по батькові, отже функція повинна адекватно опрацьовувати при передачі як трьох так і двух аргументів: прізвища та імені.

function personCollector() {

  const firstName = prompt('First name: ', 'John');

  const lastName  = prompt('Last name: ', 'Doe');

  const surName   = prompt('Surname: ',   '');

  const emptySurname = 'None';

  if(surName && true) {

    return {firstName, lastName, surName};

  } else {

    return {firstName, lastName, emptySurname};

  }

}

Шмат коду 4.1 — Функція 4

let startPersonButton = document

      .querySelector(buttonsList[index])

      .addEventListener("click", () => {

        const data = personCollector();

        if(data.surName && true) {

          alert("Person name: " + data.firstName + ", last name: " + data.lastName + ", surname: " + data.surName);

        } else {

          alert("Person name: " + data.firstName + ", last name: " + data.lastName);

        }

      });

Шмат коду 4.2 — Функція 4

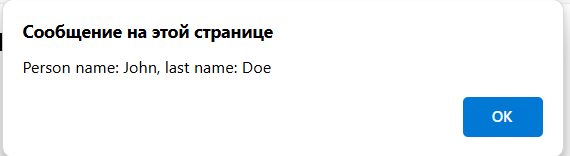


Рисунок 4.1 — Результат 1

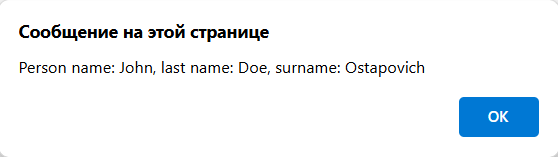


Рисунок 4.2 — Результат 2

5. Зробіть копію рішення та перетворіть function declaration в стрілочну функцію.

let startPersonLambdaButton = document

      .querySelector(buttonsList[index])

      .addEventListener("click", () => {

        const firstName = prompt('First name: ', 'John');

        const lastName  = prompt('Last name: ', 'Doe');

        const surName   = prompt('Surname: ',   '');

        const emptySurname = 'None';

        const personData = (a, b, c) => c && true ? {a, b, c} : {a, b, emptySurname};

        const personVar = personData(firstName, lastName, surName);

        if(personVar.c && true) {

          alert("Person name: " + personVar.a + ", last name: " + personVar.b + ", surname: " + personVar.c);

        } else {

          alert("Person name: " + personVar.a + ", last name: " + personVar.b);

        }

      });

Шмат коду 5 — Функція 5

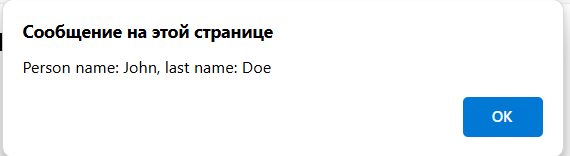


Рисунок 5.1 — Результат 1

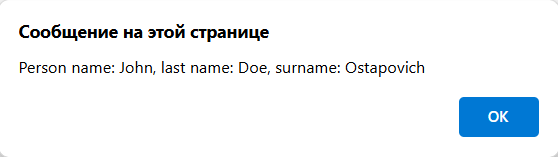


Рисунок 5.2 — Результат 2

6. Створіть функцію, яка приймає три аргументи, де перший аргумент використовується для отримання іншої функції. Реалізуйте виклик створеної функції, передавши їй 1) стрілочну функцію, що обчислює суму двох значень 2) два значення. Виконайте обчислення та відобразіть результат.

let startThreeFields = document

      .querySelector(buttonsList[index])

      .addEventListener("click", () => {

        sum = (x, y) => x + y;

        const first = prompt('First number: ', 0);

        const second = prompt('Second number: ', 0);

        alert("Result of sum of " + first + " and " + second + " is " + threeFields(sum, parseFloat(first), parseFloat(second)));

      });

Шмат коду 6.1 — Шмат функції 1

function threeFields(func, a, b)

{

  return func(a, b);

}

Шмат коду 6.2 — Шмат функції 2

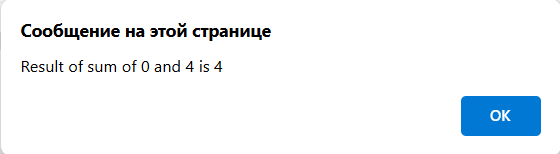


Рисунок 6 — Результат функції

7. Створіть function declaration, яка отримує довільну кількість значень та обчислює їх суму. В роботі використайте arguments.

function argumentsOverdrive(a)

{

  let sum = 0;

  for(let index = 0; index < arguments.length; index ++)

  {

    sum += parseFloat(arguments[index]);

  }

  return sum;

}

Шмат коду 7.1 — Шмат функції 1

let startArguments = document

      .querySelector(buttonsList[index])

      .addEventListener("click", () => {

        alert("Result of sum is: " + argumentsOverdrive(1,2));

        alert("Result of sum is: " + argumentsOverdrive(1));

        alert("Result of sum is: " + argumentsOverdrive(1,2,5,7));

      });

Шмат коду 7.2 — Шмат функції 2

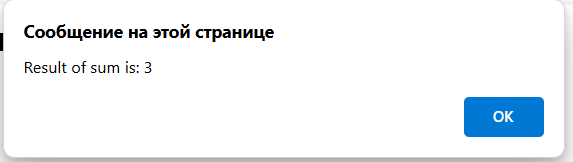


Рисунок 7.1 — Результат функції 1

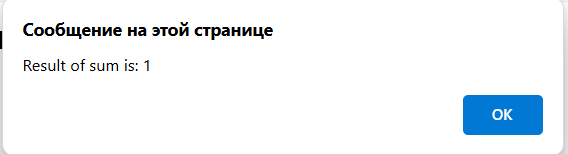


Рисунок 7.2 — Результат функції 2

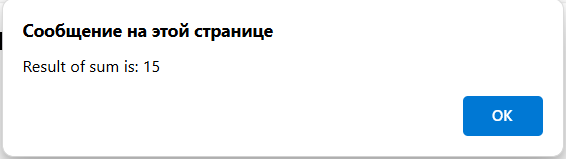


Рисунок 7.3 — Результат функції 3

8. Створіть стрілочну функцію, яка отримує довільну кількість значень та обчислює їх суму.

let startLambdaOverflow = document

      .querySelector(buttonsList[index])

      .addEventListener("click", () => {

        const lambdaSum = (...nums) => {

          let total = 0;

          for (let n of nums) total += n;

          return total;

        };

        alert("Result of sum is: " + lambdaSum(4,2));

        alert("Result of sum is: " + lambdaSum(3));

        alert("Result of sum is: " + lambdaSum(5,6,5,7));

      });

Шмат коду 8 — Функція

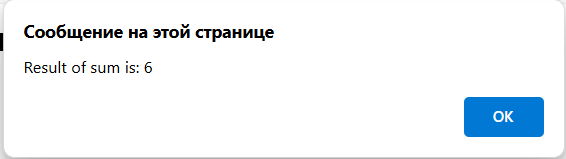


Рисунок 8.1 — Результат функції 1

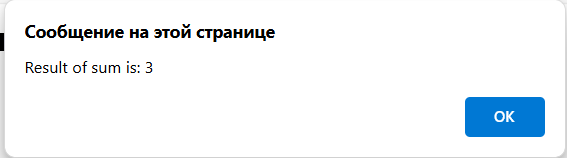


Рисунок 8.2 — Результат функції 2

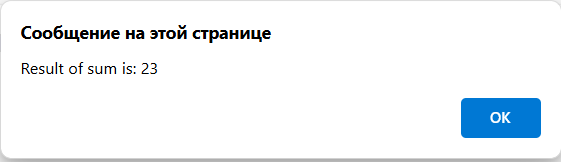


Рисунок 8.3 — Результат функції 3

9. Створіть функцію, яка повертає іншу функцію в залежності від переданого аргументу. Для спрощення, функція яка повертається повинна містити лише вивід інформації, який описує її призначення. Так, при передачі рядка ‘jpg’, повинні отримати функцію, яка формує повідомлення ‘Функція опрацювання файлів jpg’, а при передачі ‘png’ повідомлення Функція опрацювання файлів png’.

function functionReturn(input)

{

  const regularList = [

    /jpeg/,

    /jpg/,

    /png/,

    /webp/,

    /gif/,

  ];

  const labelList = [

    'jpeg',

    'jpg',

    'png',

    'webp',

    'gif',

  ]

  const output = (pattern) => alert("The program is working on " + pattern);

  for(let index = 0; index < regularList.length; index++)

  {

    if(regularList[index].test(input)) {

      return output(labelList[index]);

    }

  }

  return alert("The ptogram is working on nothing");

}

Шмат коду 9.1 — Шмат функції 1

let startFunctionReturn = document

      .querySelector(buttonsList[index])

      .addEventListener("click", () => {

        const input = prompt('Input string: ', 'example.webp');

        functionReturn(input);

      });

Шмат коду 9.2 — Шмат функції 2

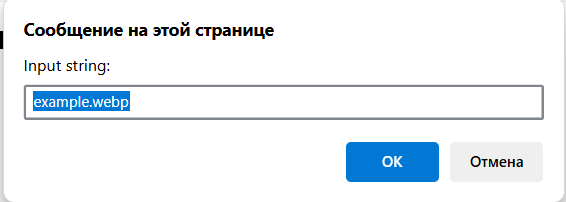


Рисунок 9.1 — Ввод данних функції 1

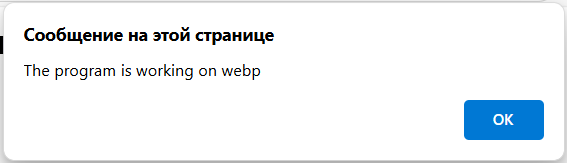


Рисунок 9.2 — Результат виконання функції 1

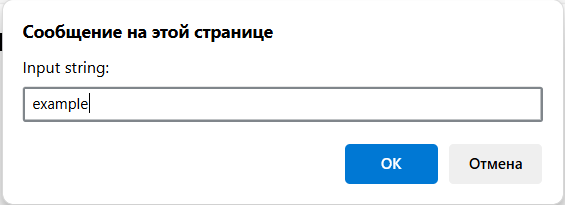


Рисунок 9.3 — Ввод данних функції 2

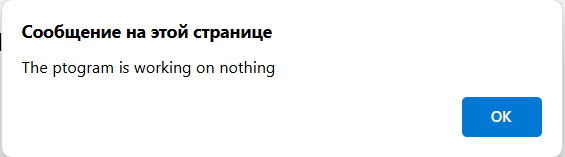


Рисунок 9.4 — Результат виконання функції 2

**Відповідь на контрольні питання:**

1. Опишіть синтаксис оголошення функції, для всіх відомих видів функцій.

Декларація функції:

function func() {

return null;

}

Функційний вираз:

let func = function() {

return null;

};

Лямбда функція:

let func = function() => true ? null;

2. Чи доступні глобальні змінні в функції?

Так, їх можна використовувати та змінювати, проте усі змінні які були оголошені у тілі функції будуть існувати у обмеженому середовищі виконання функції, без доступу ззовні.

3. Чи доступна локальна змінна функції за межами функції?

Ні, пояснив вище.

4. Який синтаксис створення функції виразу?

let func = function() {

return null;

};

5. Який синтаксис створення стрілочної функції?

let func = function() => true ? null;

6. Чи можна виконати виклик function declaration до її оголошення?

Так.

7. Чи можна виконати виклик function expression до її оголошення?

Ні, тому-що оголоситься звичайна змінна і буде викликана помилка типу, що ця зміна не є функцією.

8. Чи доступно arguments для стрілочної функції?

Ні, не доступні.

9. Назвіть функцію, яка може повернути результат функції без використання return.

Стрілочна\лямбда

10. Чи можна передати функцію у якості аргумента? Якщо можна, наведіть приклад оголошення функції та передачі їй іншої функції під час виклику, а якщо ж таке неможна виконати, обґрунтуйте причини.

Так можливо:

let test = () => true ? null;

function func(x) {

x();

}

**Висновок**

Протягом лабороторної роботи були отримані теретичні основи виконання коду сторінок та їх скриптів, особливо взаємодії із в спливаючих вікнах та формах, взаємодії із змінними та їх оголошення і т.д. Після чого було практично затверджено правильність отриманої теорії шляхом створення функціонального веб-додатку.