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

Жигальов Володимир Олександрович

Перевірив ст. викл. каф. КН

Северин М.В.

Одеса - 2025

**Лабораторна робота № 2**

**Тема:** Функції в JavaScript.

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

**Завдання**

**1.** Створити скрипт, що реалізує користувацьку функцію `min(a, b)`, яка повертає менше з двох чисел. Використати умовний тернарний оператор та \*function declaration\*.

*Рішення:*

function min1(a, b) {

return a > b ? b : a;

}

**2.** Реалізувати копію рішення завдання 1 з використанням \*function expression\*.

*Рішення:*

let min2 = function(a, b) {

return a > b ? b : a;

}

**3.** Реалізувати копію рішення завдання 2 зі зміною \*function expression\* на стрілочну функцію.

*Рішення:*

let min3 = (a, b) => a > b ? b : a;

**4.** Створити \*function declaration\*, яка приймає прізвище, ім’я та по батькові (необов'язково), і формує об’єкт з відповідними полями.

*Рішення:*

function setPerson(name, surname, fatherName = "") {

return (name + " " + surname +

(fatherName != "" ? " " + fatherName : ""));

}

**5.** Перетворити \*function declaration\* із завдання 4 у стрілочну функцію.

*Рішення:*

let setPerson2 = function(name, surname, fatherName = "") {

return (name + " " + surname +

(fatherName != "" ? " " + fatherName : ""));

}

**6.** Створити функцію, яка приймає три аргументи: функцію та два значення. Викликати її, передавши стрілочну функцію для обчислення суми двох значень та самі значення.

*Рішення:*

function useOtherFun(func, a, b) {

return func(a, b);

}

**7.** Створити \*function declaration\*, яка приймає довільну кількість значень та обчислює їхню суму. Використати `arguments`.

*Рішення:*

function sum() {

let result = 0;

for (i = 0; i < arguments.length; i++) {

result += arguments[i];

}

return result;

}

**8.** Створити стрілочну функцію, яка приймає довільну кількість значень та обчислює їхню суму.

*Рішення:*

let sum2 = (...args) => {

let result = 0;

for (i = 0; i < args.length; i++) {

result += args[i];

}

return result;

}

**9.** Створити функцію, яка повертає іншу функцію в залежності від переданого аргументу ("jpg" або "png").

*Рішення:*

function alertJpg() {

alert("функція опрацювання файлів JPG");

}

function alertPng() {

alert("функція опрацювання файлів PNG");

}

function getFuncByArg(arg) {

switch (arg) {

case "jpg":

return alertPng;

case "png":

return alertJpg;

default:

alert("такого формата не існує")

}

}

Функція для перевірки значень:

function testAllTasks() {

console.log("Task 1 (min1):", min1(5, 10)); // 5

console.log("Task 2 (min2):", min2(15, 7)); // 7

console.log("Task 3 (min3):", min3(12, 12)); // 12

console.log("Task 4 (setPerson):", setPerson("Іванов", "Іван", "Іванович")); // Іванов Іван Іванович

console.log("Task 4 (без по батькові):", setPerson("Петренко", "Петро")); // Петренко Петро

console.log("Task 5 (setPerson2):", setPerson2("Сидоренко", "Олена", "Миколаївна")); // Сидоренко Олена Миколаївна

const sumFunc = (a, b) => a + b;

console.log("Task 6 (useOtherFun):", useOtherFun(sumFunc, 8, 3)); // 11

console.log("Task 7 (sum):", sum(1, 2, 3, 4, 5)); // 15

console.log("Task 8 (sum2):", sum2(10, 20, 30)); // 60

console.log("Task 9 (getFuncByArg - jpg):");

getFuncByArg("jpg")(); // викликає alert для JPG

console.log("Task 9 (getFuncByArg - png):");

getFuncByArg("png")(); // викликає alert для PNG

console.log("Task 9 (getFuncByArg - невідомий):");

getFuncByArg("pdf"); // alert про невідомий формат

}

Результат роботи:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

**Відповідь на контрольні питання:**

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

Відповідь:

Function Declaration:

function myFunc(a, b) {

return a + b;

}

Function Expression:

const myFunc = function(a, b) {

return a + b;

};

Arrow Function:

const myFunc = (a, b) => a + b;

Function Constructor (використовується рідко):

const myFunc = new Function('a', 'b', 'return a + b');

2. Чи доступні глобальні змінні в функції?

Відповідь:

Так, функція має доступ до глобальних змінних.

Приклад:

let globalVar = 10;

function showGlobal() {

console.log(globalVar); // 10

}

3. Чи доступна локальна змінна функції за межами функції?

Відповідь:

Ні, локальні змінні недоступні за межами функції.

Приклад:

function test() {

let localVar = 5;

}

console.log(localVar); // ReferenceError

4. Який синтаксис створення функції виразу?

Відповідь:

const myFunc = function(a, b) {

return a + b;

};

5. Який синтаксис створення стрілочної функції?

Відповідь:

const myFunc = (a, b) => a + b;

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

Відповідь:

Так, завдяки "hoisting" (підняттю).

Приклад:

sayHello(); // працює

function sayHello() {

console.log("Привіт!");

}

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

Відповідь:

Ні, це призведе до помилки.

Приклад:

sayHello(); // TypeError

const sayHello = function() {

console.log("Привіт!");

};

8. Чи доступно `arguments` для стрілочної функції?

Відповідь:

Ні, стрілочні функції не мають власного об'єкта `arguments`.

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

Відповідь:

Стрілочна функція з неявним поверненням результату.

Приклад:

const sum = (a, b) => a + b;

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.

Відповідь:

Так, функцію можна передавати як аргумент.

Приклад:

function operate(func, a, b) {

return func(a, b);

}

const multiply = (x, y) => x y;

console.log(operate(multiply, 3, 4)); // 12

**Висновок**

У результаті виконання лабораторної роботи з теми «Функції в JavaScript», я набув практичного досвіду в оголошенні та використанні різних видів функцій: function declaration, function expressionта стрілочних функцій. Я засвоїв, як створювати функції з довільною кількістю аргументів, повертати функцію з функції залежно від параметра, та передавати функції як аргументи до інших функцій.

Завдяки реалізації практичних завдань, я навчився:

використовувати тернарний оператор у функціях;

працювати з параметрами за замовчуванням;

користуватись `arguments` і оператором розгортання `...` для обробки довільної кількості параметрів;

розрізняти синтаксис та особливості оголошення функцій різного типу;

перевіряти роботу функцій через консоль та логування результатів.

Цей досвід дозволив краще зрозуміти гнучкість та потужність функцій у JavaScript, що є необхідною основою для подальшого вивчення функціонального програмування та побудови складних програмних рішень.