ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет програмування та комп’ютерних

і телекомунікаційних систем

Кафедра інженерії програмного забезпечення

**Лабораторна робота № 7**

з дисципліни «Web-програмування» на тему:

Функції в JavaScript

**Виконав:**

Студент 3 курсу, групи ІПЗ-17, варіант № 15 . . С. В. Романчук

(підпис)

**Перевірив:**  . О.М. Яшина

(підпис)

Хмельницький – 2020 р.

**Тема**: Функції в JavaScript

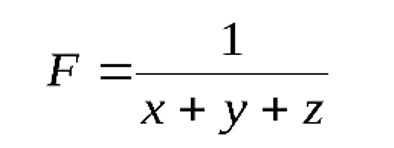
**Мета роботи**: Вивчити функції в мові програмування JavaScript та навчитись

застосовувати їх для створення динамічних сторінок

Хід роботи

**Завдання 1**: Розробити Інтернет сторінку де наряду із звичайним html текстом буде міститись прізвище, ім’я, по батькові студента, надрукованого засобами Java Script. Розмістити на сторінці кнопку, за натиском якої користувачеві на екран з’являтиметься повідомлення про дату та місце народження студента.

**Завдання 2:** Добавити сторінку на якій знаходиться лише одна кнопка. На подію натиснення призначити виконання функції розрахунку арифметичної операції:



<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8" />

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

    <title>Lab7</title>

</head>

<body onLoad="myName()">

    <p id="name"></p>

    <button onclick="birth()">Дата народження</button>

    <button onclick="calculate()">Обрахувати</button>

    <hr />

    <img src="zavd.jpg" alt="" />

    <script>

        function myName() {

            document.getElementById("name").innerHTML =

                "Романчук Сергій Володимирович";

        }

        function birth() {

            alert("17.02.1998\nНародився в місті Хмельницькому");

        }

        function calculate() {

            var x = parseInt(prompt("X", "1"));

            var y = parseInt(prompt("Y", "1"));

            var z = parseInt(prompt("Z", "1"));

            result = 1 / (x + y + z);

            alert("Результат: " + result);

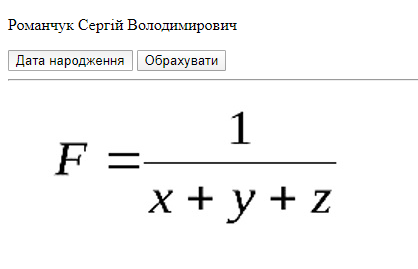
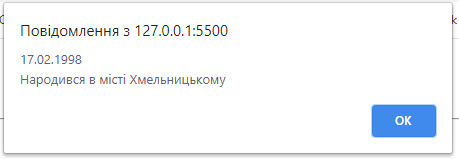
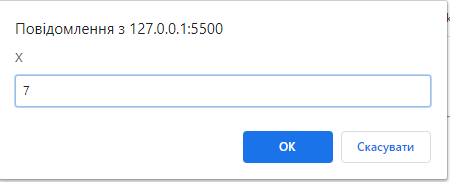
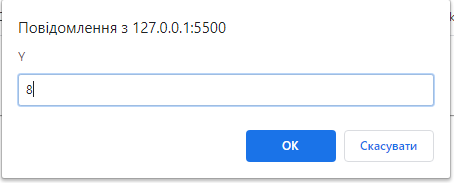
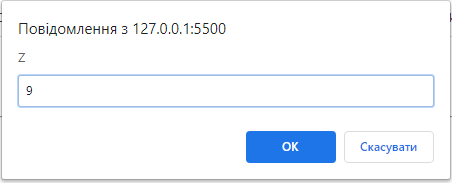
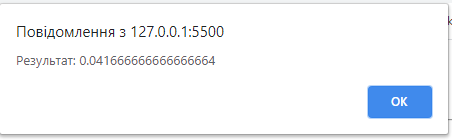
        }

    </script>

</body>

</html>

Результат:

**Контрольні питання**

**1.Функції, синтаксис, створення та виклик функції.**

Функції є одним з фундаментальних блоків у JavaScript. Функція є процедурою JavaScript, це набір команд, які виконують ту чи іншу задачу, або розраховують значення. Щоб використати функцію, ви маєте десь її визначити у тій області видимості, звідки ви бажаєте її викликати.

Визначення функції (також називається оголошенням функції, або функціональним оператором) складається з ключового слова function , після чого слідує:

* Ім'я функції.
* В дужках йде список параметрів, які передаються функції.
* Команди JavaScript, які визначають фунцію, поміщаються у фігурні дужки { }.

Для прикладу, наступний код визначає просту функцію на ім'я sum:

function sum(a, b) {

return a + b;

}

**2.Параметри функції. Область видимості змінних.**

Змінні, визначені всередині функції, недоступні ззовні цієї функції, бо змінна визначається тільки у області видимості функції. Проте, функція може звертатись до усіх змінних та функцій, визначених у області видимості, де вона оголошена. Іншими словами, функція, оголошена у глобальній області видимості, може звертатись до усіх змінних, оголошених у глобальній області видимості. Функція, оголошена всередині іншої функції, має доступ до усіх змінних, оголошених у батьківській функції, а також до будь-якої змінної, до якої має доступ батьківська функція.

Примітивні параметри (такі, як число) передаються функціям за значенням; значення передається до функції, але якщо функція змінює значення параметра, ця зміна не відображається глобально або у функції виклику.

Якщо ви передаєте об'єкт (тобто, непримітивне значення, наприклад, Array або визначений користувачем об'єкт) у якості параметра, і функція змінює властивості об'єкта, ця зміна видима за межами функції, як показано на наступному прикладі:

function func(theObject) {

theObject.make = 'Ivan';

}

var name = {make: 'Victor'};

var x, y;

x = name.make; // x отримує значення "Victor"

func(name);

y = name.make; // y отримує значення "Ivan" (властивість make була //змінена функцією)

**3. Інструкція return.**

Інструкція return визначає, яке значення повертає функція. Наприклад, наступний код поверне квадрат числа, переданого функції як параметр:

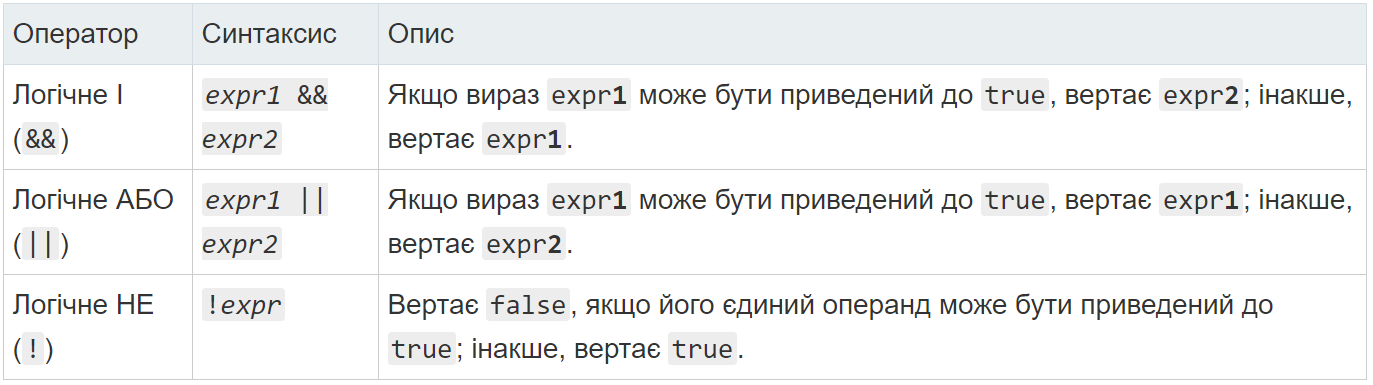
function sum(a, b) {

return a+b;

}

**4.Логічні операції у функціях.**

Логічні оператори зазвичай застосовуються до булевих (логічних) значень. В цьому випадку вони повертають значення типу Boolean. Однак, оператори && та || насправді повертають значення одного з заданих операндів, тому, якщо ці оператори використовуються зі значеннями не булевого типу, вони повернуть значення не булевого типу. Логічні оператори описані у наведеній нижче таблиці (вирази expr можуть належати до будь-якого типу, не лише булевого):



**5.Рекурсивні функції.**

Функція може посилатися на себе і викликати себе. Існує три способи рекурсивного виклику функції:

* ім'я функції
* argument.callee
* змінна в межах області, яка відноситься до функції

Наприклад, розглянемо таке визначення функції:

var func = function bar() {

// statements go here

};

У тілі функції наступні виклики еквівалентні:

bar ()

arguments.callee()

func ()

Функція, яка викликає себе, називається рекурсивною функцією. У чомусь рекурсія є аналогом циклу. Обидва виконують один і той же код кілька разів, і обидва вимагають умови (щоб уникнути нескінченного циклу, а точніше, нескінченної рекурсії в цьому випадку). Наприклад, наступний цикл:

var x = 0;

while (x < 10) { // "x < 10" є умовою циклу

// виконання логіки

x++;

}

може бути перетворений в рекурсивну функцію та виклик до цієї функції:

function loop(x) {

if (x >= 10) // "x >= 10" умова виходу (еквівалент до "!(x < 10)")

return;

// виконання логіки

loop(x + 1); // рекурсивний виклик

}

loop(0);

**Висновок**

Під час виконання лабораторної роботи я вивчив функції в мові програмування JavaScript, та навчився використовувати їх для розробки динамічних html-сторінок