Федеральное государственное автономное

Образовательное учреждение

Высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №6**

Основы работы с JavaScript

тема

Преподаватель \_\_\_\_\_\_\_\_\_ А.М. Скрипачев

подпись, дата инициалы, фамилия

Студент КИ16-16Б, №031620303 \_\_\_\_\_\_\_\_\_ В.В. Кокташев

Номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2018

# Цель работы

# Изучение основ работы с JavaScript

# Задание

1. Реализовать в виде функции поиск суммы элементов последовательности;
2. реализовать в виде функции поиск минимального элемента последовательности;
3. реализовать в виде функции поиск произведения элементов последовательности.

# Краткий теоретический материал

Язык JavaScript синтаксически близок к языкам C/C++, Java, PHP и другим C - подобным языкам. Поэтому для тех, кто знаком с такими языками не составит труда разобраться с основными языковыми конструкциями.

**Переменные**

Для объявления переменных используется ключевое слова var. Переменные можно сразу инициализировать. Можно объявить несколько переменых сразу, разделив их запятыми.

var color = "#FFF", fsize = 1024, total\_count = 0, i;

var average = null; var c = 3; d = 0; //Ошибка!

Неинициализированные переменные будут иметь неопределенное значение (undefined). Объявлять переменные можно в любом месте скрипта, но до первого обращения Типы данных переменным в javascript назначаются автоматически. Так же автоматически выполняется приведение типов.

Объявления массивов данных могут выполняться статически и динамически. Индексирование элементов начинается с нуля. Элементы массива могут быть проинициализированы при создании.

var weekdays = ["Пн", "Вт", "Ср", "Чт", "Пт"]; // статический массив из пяти элементов

// динамическое объявление массива путем создания экземпляра встроенного класса Array

var myarr; myarr = new Array(256);

myarr[0] = 255; myarr[1] = 254;

var x = myarr[7];

**Операторы**

Условный оператор if предназначен для ветвления программы в зависимости от значения (true | false) логического выражения:

if (условие) {блок операторов1} [else {блок операторов2}]

Оператор выбора switch, также, как и условный оператор, предназначен для выполнения ветвления алгоритма, но позволяет анализировать множество возможных результатов проверки условия.

Оператор break позволяет прервать выполнение оператора, если его не указать, то будут выполнены все последующие операторы.

switch (условие) {

case значение1 : {блок операторов1; break;}

case значение2 : {блок операторов2; break;}

case значение3 : {блок операторов3; break;} ...

[default : {блок операторов по умолчанию};]

}

Цикл со счетчиком for. Используется для циклов с заданным числом итераций (примечание: на самом деле конструкция for может использоваться и для построения любых циклов, все зависит от того, как и какие значения указаны в качестве параметров цикла).

for ([начальное значение]; [условие]; [приращение]) {блок операторов;}

Цикл с постусловием do...while. Выполняется, пока условие является истинным. Всегда выполняется хотя бы один раз.

do {блок операторов;} while (условие)

Цикл с предусловием while. Выполняется, если условие является истинным. Может не выполниться ни разу.

while (условие) {блок операторов;}

Операторы break и continue - используются для прерывания выполнения цикла или завершения текущей итерации.

Поэлементный цикл for (... in ...) применяется для выполнения команд над каждым элементом массива или коллекции.

for (переменная in массив|объект|коллекция) {блок операторов;}

Функции. Представляют возможность создания повторно используемого кода. Функция может принимать параметры и возвращать значение в вызывающую программу. Если в функции не предусмотрен возврат значения, то она работает как процедура.

function имяФункции ([список параметров]) {блок операторов; [return возвращаемоеЗначение;]}

**Объект Math**

Объект Math представляет математические константы и функции. Константы представлены свойствами объекта, а функции - его методами. Их назначение понятно из названий:

Свойства: E, LN2, LN10, LOG2E, LOG10E, PI, SQRT1\_2, SQRT2.

Методы: abs, acos, asin, atan, ceil, cos, exp, floor, log, max, min, pow, random, round, sin, sqrt, tan.

Примеры использования:

var r = 1.8, theta = 30, a, x, y, D;

var rnd = Math.round(Math.random()\*99)+1;

D = Math.PI\*r\*r; x = Math.max(1,7,5,9);

y = Math.pow(2,10);

with (Math) { y = r\*sin(theta); x = r\*cos(theta); }

**Объект string**

Встроенный объект string представляет литерал (строку символов), заключенный в одинарные или двойные кавычки или вычислимое выражение, которое может быть интерпретировано как строка. Для объекта string определены следующие свойства и методы:

Свойства: length (длина строки).

Методы (не все): anchor (якорь), bold (полужирное начертание), charAt (символ в позиции), fixed (преформат), fontcolor (цвет шрифта), fontsize (размер шрифта), indexOf (индекс первого вхождения символа), italics (курсив), link (гиперссылка), substring (подстрока), toLowerCase (строчные), toUpperCase (прописные).

Несколько примеров использования объекта string:

var hello = "Hello, ", w = "World!"; var str = hello + w; // конкатенация строк

document.write(str.bold());

document.write(str.toUpperCase());

document.write(hello.fontsize(6));

document.write(hello.substring(0,3));

document.write(hello.link("http://localhost"));

document.write(w.indexOf("l"));

alert("string lehgth = " + str.length);

# Ход работы

1. Была создана страница Lab6.html, к которой подключен файл scripts.js (рисунок 1):



Рисунок 1 – Подключение javascript файла

1. Разработаны функции поиска суммы четных элементов последовательности, суммы модулей элементов последовательности, а также разницы максимального и минимального элементов последовательности. Тестовый пример работы этих функций представлен на рисунке 2.

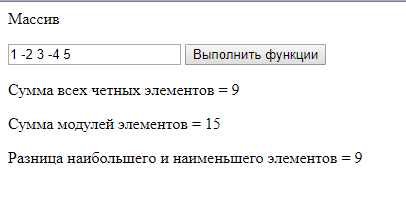


Рисунок 2 – Результат работы функций

# Вывод

В ходе лабораторной работы были изучены основы работы с JavaScript и разработаны три функции, закрепляющие полученные знания на практике.

# ПРИЛОЖЕНИЕ

Листинг 1 - файл Lab6.html

<html>

<head>

<meta charset="utf-8">

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

<script src="scripts.js"></script>

<title>Lab6</title>

</head>

<body>

<p>Массив</p>

<input type="text" id="masIn">

<button onclick="ButtonClick()" type="button">Выполнить функции</button>

<p id="outEvenSum">Сумма всеx четныx элементов</p>

<p id="outModuleSum">Сумма модулей элементов</p>

<p id="outMax\_min">Разница наибольшего и наименьшего элементов</p>

</body>

</html>

Листинг 2 – файл scripts.js

function ButtonClick()

{

var evenOut= document.getElementById("outEvenSum"),

moduleOut = document.getElementById("outModuleSum"),

maxOut = document.getElementById("outMax\_min");

var text = document.getElementById("masIn").value.trim().split(" ");

var mas = new Array();

for (var i = 0; i < text.length; i++)

if (!isNaN(parseInt(text[i])))

mas[i] = parseInt(text[i]);

evenOut.textContent = "Сумма всеx четныx элементов = " + evenSum(mas);

moduleOut.textContent = "Сумма модулей элементов = " + moduleSum(mas)

maxOut.textContent = "Разница наибольшего и наименьшего элементов = " + max\_min(mas);

}

function evenSum (mas)

{

var sum = 0;

for (var i = 0; i < mas.length; i += 2) {

sum += mas[i];

}

return sum;

}

function moduleSum (mas)

{

var sum = 0;

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

if (mas[i] > 0)

sum += mas[i];

else

sum -= mas[i];

}

return sum;

}

function max\_min (mas)

{

var difference = Math.max.apply(null, mas) - Math.min.apply(null, mas);

return difference;

}