# Лабораторная работа № 3 Использование функций в JavaScript

**Цель работы:** научиться использовать стандартные и создавать пользовательские функции в JavaScript.

**Краткие теоретические сведения:**

Функции представляют собой набор инструкций, выполняющих определенное действие или вычисляющих определенное значение.

Синтаксис определения функции:

function имя\_функции([параметр [, ...]]){

// Инструкции

}

Определение функции начинается с ключевого слова function, после которого следует имя функции. Наименование функции подчиняется тем же правилам, что и наименование переменной: оно может содержать только цифры, буквы, символы подчеркивания и доллара ($) и должно начинаться с буквы, символа подчеркивания или доллара.

После имени функции в скобках идет перечисление параметров. Даже если параметров у функции нет, то просто идут пустые скобки. Затем в фигурных скобках идет тело функции, содержащее набор инструкций.

Чтобы выполнить тело функции, необходимо вызвать функцию с передачей параметров или без параметров:

function sum(a, b, c){

let d = a + b + c;

console.log(d);

}

sum(1, 2, 3);

sum();

В отличии от компилируемых языков программирование, где количество фактических параметров, передаваемых при вызове функции, и количество формальных параметров, описанных при объявлении, должны совпадать, JavaScript позволяет не соблюдать это правило. Если для параметров не передается значение, то по умолчанию они имеют значение "undefined".

Для определения количества переданных параметров и доступа к их значениям при вызове функции создается массив arguments, доступный внутри тела функции.

Есть способ определения значения для параметров по умолчанию:

function sum(a = 1, b = 2, c = 3){

let d = a + b + c;

console.log(d);

}

С помощью spread-оператора мы можем указать, что с помощью параметра можно передать переменное количество значений:

function display(season, ...temps){

console.log(season);

for(index in temps){

console.log(temps[index]);

}

}

display("Весна", -2, -3, 4, 2, 5);

display("Лето", 20, 23, 31);

Функция может возвращать результат. Для этого используется оператор return:

function sum(a, b){

return a + b;

}

console.log(sum(1, 2);

Функции могут выступать в качестве параметров других функций.

JavaScript позволяет определять функции внутри функций и возвращать результат в виде функции.

Среди функций отдельно можно выделить рекурсивные функции. Их суть состоит в том, что функция вызывает саму себя.

Рассмотрим, например, функцию, определяющую факториал числа:

function getFactorial(n){

if (n === 1){

return 1;

}

else{

return n \* getFactorial(n - 1);

}

}

let result = getFactorial(4);

console.log(result); // 24

**Практическая часть:**

**Использование встроенных функций**

**Задание 1.**

1. Создайте файлы z3\_11.html и z3\_12.html, демонстрирующие использование встроенных функций объекта Date:

1.1

<script>

"use strict";

let today = new Date();

let hours = today.getHours();

let minute = today.getMinutes();

if (minute < 10) minute = "0" + minute;

if (hours < 12) {

var time12 = hours + ":" + minute + " am";

} else {

var time12 = hours - 12 + ":" + minute + " pm";

}

console.log("текущее время - ", time12);

</script>

1.2

<script>

"use strict";

let today = new Date();

let newYearDay = new Date(today.getFullYear() + 1, 0, 1);

const day = 86400000;

const hour = 3600000;

const minute = 60000;

let time = newYearDay.getTime() - today.getTime();

let days = Math.floor(time / day);

let hours = Math.floor((time - days \* day) / hour);

let minutes = Math.floor((time - days \* day - hours \* hour) / minute);

let str = days + " дней, " + hours + " часов, " + minutes + " минут.";

console.log("До Нового года осталось ", str);

</script>

**Создание пользовательских функций**

**Задание 2.**

Создайте файл z3\_2.html, демонстрирующий создание пользовательской функции с аргументами и обращение к ней из основной части скрипта:

<script>

"use strict";

//Описание фvнкции вывода строки с переходом на новую

function PrintBR(txt) {

document.write(txt, "<br />");

}

//Обрашение к функции с передачей фактических параметров

PrintBR("Это первая строка");

PrintBR("Это вторая строка");

PrintBR("Это еще одна строка");

</script>

**Задание 3.**

Создайте файл z2\_31.html, демонстрирующий создание пользовательской функции, возвращающей значение:

<script>

"use strict";

function AddNums(firstnum, secondnum) {

return firstnum + secondnum;

}

document.write("3 + 5 = ", AddNums(3, 5));

let a = 5;

let b = 25;

document.write("<br>", a, " + ", b, " = ", AddNums(a, b));

</script>

**Задание 4.**

Создайте файл z3\_4.html, демонстрирующий использование функции с необязательными параметрами и параметрами по умолчанию:

<script>

"use strict";

function printText(txt, size = 14) {

document.write(`<div style="font-size:${size}pt">${txt}</div>`);

}

printText("<p>Крупный шрифт", 32);

printText("<p>Шрифт по умолчанию - первая строка");

printText("<p>Шрифт по умолчанию - вторая строка");

printText("<p>Мелкий шрифт", 8);

</script>

**Задание 5.**

Создайте файл z3\_5.html, демонстрирующий передачу функцию в качестве аргумента другой функции:

<script>

"use strict";

function kvadrat(a) {

return a \* a;

}

function polinom(a, k) {

return k(a) + a + 5;

}

let result = polinom(3, kvadrat);

document.write("<h2>result=", result);

</script>

**Задание 6.**

Создайте файл z3\_6.html, демонстрирующий использование функции как переменной:

<script>

"use strict";

let i = 5;

function f(a, b, c) {

if (a > b) return c;

}

document.write("Значение переменной i: " + i.valueOf());

document.write("<p>Значение переменной f:<BR>" + f.valueOf());

</script>

**Задание 7.**

Создайте файл z3\_7.html, демонстрирующий использование массива arguments:

<script>

function mean(a, b) {

if (arguments.length > mean.length) {

alert("Аргументов больше, чем надо");

}

var result = 0;

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

result += arguments[i];

}

return result / arguments.length;

}

document.write("Среднее - ", mean(2, 3), "<br>");

document.write("Среднее - ", mean(2, 4, 6));

</script>

**Индивидуальные задания**

Во всех скриптах, в заголовке окна браузера должны быть ваши фамилия и имя!!!

**Задание 1.**

В скрипте lab3-1.html, используя аргументы size, day и color функции weekDay(), отобразите названия дней недели уменьшающимся размером (начиная с 7) и разными цветами:

понедельник

вторник

среда

четверг

пятница

суббота

воскресенье

**Задание 2.**

В соответствии со своим вариантом написать скрипт для вычисления значения функции b=f(x,y,z). Значения x, y и z должны вводиться пользователем, используя метод prompt. При выводе информации предусмотреть форматирование документа, вывод текста задания, включая рисунок исходной функции, и вывод информации о разработчике скрипта.

| Вариант | Вид функции | Вариант | Вид функции |
| --- | --- | --- | --- |
| 1 |  | 16 |  |
| 2 |  | 17 |  |
| 3 |  | 18 |  |
| 4 |  | 19 |  |
| 5 |  | 20 |  |
| 6 |  | 21 |  |
| 7 |  | 22 |  |
| 8 |  | 23 |  |
| 9 |  | 24 |  |
| 10 |  | 25 |  |
| 11 |  | 26 |  |
| 12 |  | 27 |  |
| 13 |  | 28 |  |
| 14 |  | 29 |  |
| 15 |  | 30 |  |

**Задание 3.**

В соответствии со своим вариантом вычислить значение функции с помощью разложения в ряд Тейлора. Задание реализовать с использованием рекурсии. Вывести на экран значения функции, используя стандартные функции и с помощью разложения функции в ряд Тейлора.

|  |  |
| --- | --- |
|  | Разложение функции в ряд Тейлора |
| 1. |  |
| 2. |  |
| 3. |  |
| 4. |  |
| 5. |  |
| 6. |  |
| 7. |  |
| 8. |  |
| 9. |  |
| 10. |  |
| 11. |  |
| 12 |  |
| 13. |  |
| 14. |  |
| 15. |  |

**Требование по содержанию отчета:**

В отчете должны быть отображены следующие пункты:

1. Титульный лист.

2. Цель работы.

3. Задания.

5. Листинг кода сценария с комментариями.

6. Результат выполнения скрипта.

8. Выводы.

**Контрольные вопросы для защиты:**

1. Что представляет из себя функция?
2. Как создать пользовательскую функцию?
3. Что такое формальные и фактические параметры?
4. Чем отличается описание функций, не возвращающих и возвращающих значения?
5. В чем особенность использования функции как переменной?
6. В чем особенность использования функции как объект?
7. Для чего используется функция eval?
8. Чем отличаются глобальные и локальные переменные?
9. Как задать значение параметра функции по умолчанию?
10. Для чего используется массив arguments?