JS

Лабораторная работа 2.

Создадим папку lr2. Откроем ее в vsCode (файл – открыть папку…). Это будет проектом. Создадим в нем файлы Index.html и script.js.

Index.html

<html>

<head>

<meta charset="UTF-8">

    <script src="script.js"></script>

<title>Лабораторная работа 2</title>

</head>

<body>

</body>

</html>

**Выполнение.**

1. Массивы.
   1. Массивы – упорядоченный набор значений. Массивы в js динамические.

Массив объявляется аналогично переменным (var, let, const).

let arr = [1, 2, 3, 4, 5];

let arr1 = ['hello', 'world', 1, 2, 3, true];

Либо можно создать новый объект типа Array.

let arr = new Array(1,2,3,4,5);

Массив может содержать пустые ячейки (в них будет храниться undefined)

let arr2 = [5,,4];

Доступ к элементам массива может осуществляться по индексу (индексация от 0).

let arr = [1, 2, 3, 4, 5];

arr[1] = 20;

Элементы могут обозначаться не только индексам, но и ключами *(в этом случае массив ведет себя, как объект, а эти элементы – его свойства).*

let arr = [1, 2, 3, 4, 5];

arr['someKey'] = 'someValue';

console.log(arr);



Для того, чтобы узнать длину массива – нужно обратиться к его свойству length.

let arr = [1, 2, 3, 4, 5];

console.log(arr.length);

* 1. Перебор элементов массива. Цикл for in

let arr = [1, 2, 3, 4, 5];

arr['one'] = 'smth';

for(let i in arr) {

    console.log(arr[i]);

}

И for of

let arr = [1, 2, 3, 4, 5];

for(let i of arr) {

console.log(i);

}

* 1. Многомерные массивы в js

JS не поддерживает многомерные массивы, но в ячейках одномерного массива могут содержаться одномерные массивы.

let arr = [1, 2,['Hello', ['foo', 'bar'],'world'], 3, 4, 5];

console.log(arr[2][1][0]);

* 1. Методы для работы с массивами

*В js есть глобальные объекты, один из них – document (объект для работы с документом). Для удобства будем выводить результаты работы скриптов при помощи его метода write(string), который выводит строку после тега body.*

**arr.toString()**

Данный метод преобразует массив в строку (при выводе массива применяется автоматически)

let arr = [1, 2, 3, 4, 5];

document.write(arr.toString());

**arr.join(string)**

Данный метод преобразует массив в строку, разделитель между элементами - string

let arr = [1, 2, 3, 4, 5];

document.write(arr.join('<br>'));

**arr.reverse()**

Данный метод реверсирует массив

**arr.sort()**

Данный метод сортирует элементы массива (по умолчанию по возрастанию строки).

let arr = [1, 11, 2, 1110, 222];

document.write(arr.sort().toString());

1,11,1110,2,222

Метод может принимать в параметры функцию (callback с двумя параметрами *a* и *b*), определяющую правила сортировки. Если функция возвращает -1, то *a* должен стоять перед *b* и наоборот, если возвращает 1.

let arr = [1, 11, 2, 1110, 222];

arr.sort(function howToSort(a, b) {

    return (a > b) ? 1 : -1;

});

document.write(arr.toString());

**arr.concat(value1, value2, … valueN)**

Данный метод создаёт новый массив, в который копируются элементы из arr, а также value1, value2, ... valueN.

**arr.slice(begin, end)**

Данный метод копирует участок массива от begin до end(не включительно), не включая end. Исходный массив при этом не меняется. Если в параметрах указать отрицательные значения для begin и end, то элементы будут считаться с конца массива.

let arr = [1, 2, 3, 4, 5]

document.write(arr.slice(1, 4).toString());

**arr.splice(index, deleteCount, elem1, ..., elemN)**

Данный метод удаляет deleteCount элементов, начиная с номера index, а затем вставить elem1, ..., elemN на их место. Возвращает массив из удалённых элементов.

let arr = [1, 1, 5]

console.log(arr.splice(1, 1, 2, 3, 4));

document.write(arr.toString());

**arr.push(value1,value2,…valueN)**

Данный метод добавляет в конец массива arr значения, указанные в параметрах. Метод возвращает новую длину массива.

**arr.pop()**

Данный метод удаляет последний элемент массива arr. Метод возвращает удаленный элемент.

**arr.unshift(value)**

Данный метод добавляет в начало массива value. Метод возвращает новую длину массива.

**arr.shift()**

Данный метод удаляет первый элемент массива. Метод возвращает удаленный элемент.

**delete arr[index]**

Оператор delete удалит значение ячейки массива и сделает ее undefined.

**index in arr**

Так можно проверить су ществует ли в массиве элемент с индексом Index

let arr = [1, 2, 3, 4, 5]

if (2 in arr) {

    alert('существует');

}

**Array.isArray(arr)**

Данный метод проверяет содержится ли массив в arr.

1. Строки

2.1 Строки являются объектами типа string.

Создать строковую переменную можно так

let str = 'Hello world';

или так

let str = new String('Hello world');

Свойство length характеризует ее длину.

let str = 'Hello world';

console.log(str.length);

2.2 Методы работы со строками

**str.charAt(index)**

Данный метод возвращает символ с индексом index. Аналогично можно str[index].

**str.concat(string)**

Возвращает конкатенацию строк str и string.

**str.indexOf(substr)**

Данный метод ищет место вхождения подстроки в строку. Возвращает индекс первого вхождения подстроки в строку.

**str.lastIndexOf(substr)**

Данный метод ищет место вхождения подстроки в строку. Возвращает индекс последнего вхождения подстроки в строку.

**str.split(dev)**

Данный метод возвращает массив из строки, разделитель – dev.

let str = 'a-b-c-d-e-f';

let arr = str.split('-');

console.log(arr);

**str.replace(substr, substr2)**

Данный метод находит первое вхождение substr в str и меняет ее на substr2. Может принимать колбэк с правилом замены.

alert('10 13 21 48 52'.replace(/\d+/g, function(match) {

    return parseInt(match) < 30 ? '\*' : match;

}));

**str.substr(begin, length)**

Данный метод возвращает подстроку из str, которая находится от позиции begin длинной lenght.

**str.substring(begin, end)**

Данный метод возвращает подстроку из str, которая находится между индексами begin и end(не включительно).

**str.toUpperCase()**

**str.toLowerCase()**

Перевод строки в верхний и нижний регистр.

**str.slice(begin, end)**

Аналогично методу substring, но можно использовать отрицательные индексы, тогда отсчет с конца.

**str.search(string)**

Поиск в строке по регулярному выражению string. Вернет индекс начала вхождения.

let str = 'Hello 123456 world';

console.log(str.search(/[0-9]/));

**str.match(string)**

Поиск в строке по регулярному выражению string. Вернет массив с искомой подстрокой.

let str = 'Hello 123456 world';

console.log(str.match(/[0-9]/));



**str.bold()**

Оборачивает строку тегами <b></b>

**str.italics()**

Оборачивает строку тегами <i></i>

**str.link(lnk)**

Оборачивает строку тегами <a href = “lnk”></a>

**str.blink()**

Оборачивает строку тегами <blink></blink>

**str.fontColor(color)**

Оборачивает строку тегами <font color="color"></font>

**str.fontSize(size)**

Оборачивает строку тегами <font size="size"></font>