Лабораторная работа 4

Основы JavaScript

Цель: освоить базовый синтаксис javascript

- Лабораторная работа сдается преподавателю лично студентом.
- Сдача лабораторной работы заключается в том, что студент отвечает на вопросы преподавателя по заданиям, а также показывает и комментирует результаты самостоятельно выполненных заданий.
- Всё что написано в заданиях «изучите, узнайте и прочее» нужно изучить и для себя зафиксировать, так как преподаватель может спросить об этом при сдаче лабораторной работы.

(!) В качестве онлайн-справочников можно пользоваться http://javascript.ru/ или https://learn.javascript.ru/

(!) В качестве редактора кода можно воспользоваться Visual Studio или Code Studio (Создать файл, выбрать нужный тип .html или .js) или ресурсом https://jsfiddle.net/

ЧАСТЬ 1. Переменные

Задание 1.1. Вспомните особенности поведения **var** и **let**.

a) Запустите следующий код. Объясните результат, почему **var** работает иначе, чем **let?**

```
function testVar() {
  if (true) {
    var x = 10;
  }
  console.log(x); // Что выведет?
}

function testLet() {
  if (true) {
    let y = 20;
  }
  console.log(y); // Что выведет?
}

testVar();
testLet();
```

b) Выполните следующий код и объясните, почему **var** и **let** позволяют переназначать значения, а **const** — нет. Добавьте пример объекта с **const** и покажите, что его содержимое можно изменять

```
var a = 10;
let b = 20;
const c = 30;

a = 15; // Что произойдёт?
b = 25; // Что произойдёт?
c = 35; // Что произойдёт?

console.log(a, b, c);
```

c) Запустите следующий код. Объясните разницу в поведении **var** и **let** при использовании до их объявления

```
console.log(x); // Что выведет?
var x = 10;

console.log(y); // Что выведет?
let y = 20
```

Задание 1.2. Создайте следующие ниже таблицы посредством JavaScript-кода и закрасьте их ячейки в соответствующие цвета. Можно воспользоваться методом **write()** объекта **document**. Метод **document.write()** используется для записи текста или HTML-кода непосредственно в документ. Метод добавляет содержимое в текущую точку HTML-документа, где находится парсер (обработчик).

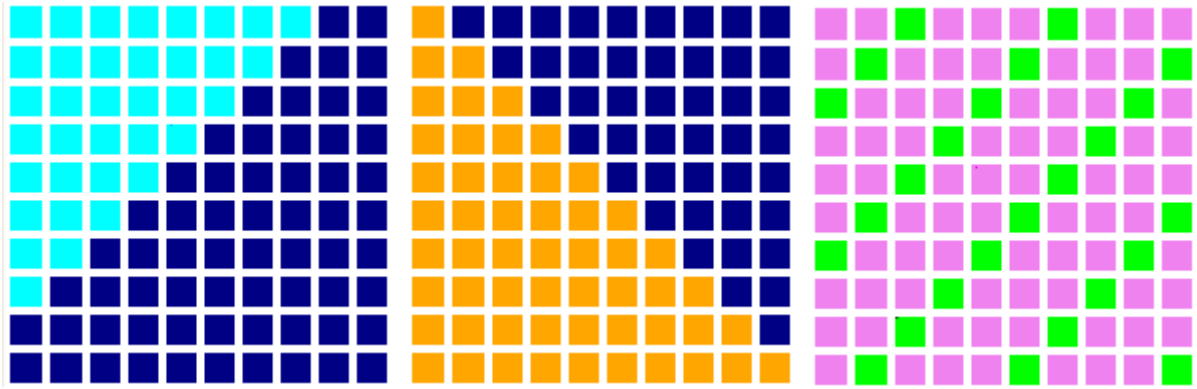
document.write(строка);

где **строка** - текст или HTML-код, который нужно записать в документе.

Пример:

Код	Отображение
Первый параграф <script> document.write(' Второй параграф '); </script> Третий параграф	Первый параграф Второй параграф Третий параграф

Метод является устаревшим и может замедлить загрузку страницы. Альтернативой является использование современных методов работы с DOM.



Задание 1.3. Дан массив целых чисел. Напишите код, который возвращает ***true***, если в массиве встречаются дубликаты, и ***false*** – в обратном случае. Ввод входного массива сделайте через функцию prompt().

Задание 1.4. Палиндромы в ДНК — это участки, которые читаются одинаково в обоих направлениях (например, "ATGCAT"). Напишите программу, которая находит все палиндромные подстроки заданной длины в последовательности ДНК.

Задание 1.5. Напишите код для сравнения двух последовательностей ДНК и выявления мутаций, то есть различий между ними. Для сравнения последовательностей используйте расстояние Левенштейна.

Задание 1.6. Дан массив объектов, где каждый объект описывает человека и его друзей. Каждый объект имеет следующую структуру:

```
{
  name: "Имя", // Имя человека
  friends: ["Имя друга 1", "Имя друга 2", ...] // Массив имён друзей
```

}

Напишите функцию, которая определяет по веденным двум именам и числу n обладают ли два человека дружбой в n степени (по аналогии с "n-м рукопожатием"). Степень дружбы определяется как минимальное количество "связей" (друзей) между двумя людьми. Функция возвращает ***true***, если связь найдена за указанное n или меньшее число шагов. И возвращает ***false***, если общих друзей нет за указанное число шагов.

Задание 1.7. Задается месяц и год, требуется вывести количество рабочих дней в этом месяце. День считается рабочим, если он не выходной и не праздничный. Праздничные дни задаются массивом (число и месяц), а выходные дни – субботы и воскресенья.

Задание 1.8. Компания занимается доставкой товаров. Задается список городов cities, между которыми можно перемещаться, и расстояния между ними. Также есть список заказов orders, каждый из которых содержит информацию о городе отправления, городе назначения и весе груза.
Необходимо написать функцию, которая:

1. Определяет, существует ли маршрут между городами отправления и назначения для каждого заказа.
2. Если маршрут существует, вычисляет общее расстояние для доставки всех заказов.
3. Если маршрут не существует, возвращает список заказов, которые невозможно доставить.

Функция должна возвращать объект с двумя полями:

- totalDistance: общее расстояние для доставки всех возможных заказов.
- undeliverableOrders: массив заказов, которые невозможно доставить.

(!) Обратите внимание, что каждая посылка перевозится отдельно.

Пример входных данных:

```
const cities = [
  { from: "Москва", to: "Санкт-Петербург", distance: 700 },
  { from: "Москва", to: "Нижний Новгород", distance: 400 },
  { from: "Санкт-Петербург", to: "Нижний Новгород", distance: 900 },
  { from: "Нижний Новгород", to: "Казань", distance: 300 }
];
const orders = [
  { id: 1, from: "Москва", to: "Нижний Новгород", weight: 50 },
  { id: 2, from: "Москва", to: "Казань", weight: 30 },
  { id: 3, from: "Санкт-Петербург", to: "Казань", weight: 20 }
];
```

Ожидаемый результат:

```
{
  totalDistance: 2300, // Москва → Нижний Новгород (400) + Москва → Казань (через Нижний Новгород, 400 + 300) + Санкт-Петербург → Казань (через Нижний Новгород, 900 + 300)
  undeliverableOrders: []
}
```

ЧАСТЬ 2. Массивы и их методы

Задача 2.1. Напишите функцию `filterByRange(arr, a, b)`, которая принимает массив `arr`, ищет элементы со значениями больше или равными `a` и меньше или равными `b` и возвращает результат в виде массива. **Функция должна возвращать новый массив и не изменять исходный**.

Задача 2.2. Напишите функцию `filterByRange2(arr, a, b)`, которая принимает массив `arr` и удаляет из него все значения кроме тех, которые находятся между `a` и `b`, алогично как в предыдущей задаче, но **функция должна изменять принимаемый массив и ничего не возвращать**.

Задача 2.3. Есть массив объектов с полями name, age, from.

a) Напишите функцию, которая вычисляет среднее значение поля age.

b) Напишите функцию, которая возвращает значения поля name объектов массива, у которых значение поля age больше среднего значения age, вычисленного по всему массиву.

c) Напишите функцию, которая возвращает значение from, наиболее часто встречающееся в массиве, и соответствующие значения поля name (те, у кого from равен этому наиболее часто встречающемуся значение).

d) Напишите функцию, которая возвращает список объектов, отсортированный по полю, которое задает пользователь. Если пользователь указал неверное значение поля, то должно выводится сообщение об этом.

Пример массива (можно скопировать)

```
[
      {"name": "Alex", "age":30, "from": "Moscow"},
      {"name": "Sofa", "age":ы22, "from": "Kazan"},
      {"name": "Ilya", "age":26, "from": "Piter"},
      {"name": "Kolya", "age":30, "from": "Kirov"},
      {"name": "Alex", "age":18, "from": "Piter"},
      {"name": "Boris", "age":38, "from": "Kirov"},
      {"name": "Olga", "age":16, "from": "Kirov"},
      {"name": "Alex", "age":42, "from": "Moscow"},
]
```

Задача 2.4. Дан массив (можно скопировать):

```
[{
      model: "Bert",
      architecture: "encoder",
      application: ["classification", "masked prediction", "embedding"]
},
{
      model: "GPT",
      architecture: "decoder",
      application:  ["text  generation",  "question  answering",  "instructed
      generation", "classification"]
},
{
      model: "T5",
      architecture: "encoder-decoder",
      application: ["classification", "summarization", "question answering"]
}]
```

Требуется пробежаться по всем элементам массива и создать новый массив из строковых значений, которые встречаются в поле `application`, **исключая дублирование значений**.

То есть результатом должен быть массив (порядок элементов неважен):

```
["classification", "masked prediction", "embedding", "text generation", "question
answering", "instructed generation", "summarization"]
```

Задание 2.5. Дан массив (можно скопировать)

```
      [{
            name: "Alice",
            code: 5005,
            home: {roof:"yellow", door:"yellow", wall:"yellow"}
      },
      {
            name: "Bob",
            code: 5105,
            home: {roof:"green", door:"grey", wall:"grey"}
      },
      {
```

```
        name: "Carol",
        code: 5015,
        home: {roof:"yellow", door:"white", wall:"yellow"}
    },
    {
        name: "Dave",
        code: 5115,
        home: {roof:"pink", door:"red", wall:"yellow"}
    }]
```

Требуется сформировать массив из значений поля `code` тех объектов, у которых в поле `home` объект имеет два одинаковых цвета (то есть совпадают минимум два цвета из полей `roof`, `door` и `wall`).

Задание 2.6. Дан массив аналогичный массиву из предыдущего задания, удалите из массива элементы, у которых в поле `home` встречается цвет, заданный пользователем (через диалоговое окно). Если цвета не будет, то требуется вывести окно с сообщением, что такого объекта в массиве нет.

Задание 2.7. Пусть есть массив чисел или строк. Например:

```
const data = [1, 2, 3, 2, 4, 2, 5, 3, 3, 3];
```

Найдите элемент, который встречается чаще всего. Если таких элементов несколько, то выведите их все в виде списка, где каждый элемент списка содержит значение и свои индексы в исходном массиве.

Задание 2.8. Пусть есть массив объектов, представляющих фильмы. Каждый объект содержит следующие свойства:

`title` (название фильма),
`genre` (жанр фильма, например, "Драма", "Комедия", "Боевик"),
`rating` (рейтинг фильма от 1 до 10).

Выполните:

a) Отсортировать фильмы по убыванию рейтинга.

b) Создать новый массив, содержащий только фильмы с рейтингом выше 7.

c) Сгруппировать фильмы по жанрам.