**ЛАБОРАТОРНАЯ РАБОТА**

по дисциплине «Информационные технологии в таможенном деле»

для студентов специальности 1-96 01 01 «Таможенное дело»

**ОСНОВЫ РАБОТЫ С МАССИВАМИ ПРИ ПОМОЩИ JavaScript**

**Цель работы:** изучить основные принципы программирования на языке Javascript, рассмотреть способы взаимодействия с массивами, научиться создавать их на основе шаблона HTML-документа, а также с помощью каскадных таблиц стилей CSS.

**Введение в JavaScript**

JavaScript — основной язык программирования в интернете. Он позволяет расширить стандартно поведение пользователя на веб-страницах. С JavaScript страницы из сухих и статичных, которые просто показываются друг за другом, становятся интерактивными и взаимодействуют пользователями, откликаются на их действия. Можно получать и применять данные из Сети и делать многое другое.

Понятие **DOM**

При работе на стороне клиента можно разделить рабочие объекты на две сущности: **JavaScript** и **HTML**. Первая отвечает за динамическую логику, вторая — за разметку. Их взаимодействие происходит через специальный объект — объектную модель документа (DOM — **D**ocument **O**bject **M**odel). DOM генерируется в момент загрузки HTML-документа браузером.

1. В момент загрузки браузер читает разметку HTML.
2. Во время чтения создается набор сущностей, которые связаны между собой на основании верстки.
3. Сущности сохраняются в модели DOM.
4. Код на JS общается с DOM, обращаясь к элементам верстки и их содержимому.
5. Когда код меняет структуру DOM, браузер автоматически обновляет (но не перезагружает) страницу, показывая новую структуру и содержимое.

Принципы написания кода на **JavaScript:**

1. Страница создается по стандартной схеме. Верстается структура HTML, к ней применяются стили и добавляется содержимое.
2. К странице подключается код JavaScript. На примере вы увидите, что это происходит так же, как подключение файла CSS — код можно вынести в отдельный файл. Но как и с CSS, никто не запрещает писать JavaScript-код внутри разметки страницы.
3. При загрузке страницы браузером происходит построение DOM-модели. Браузер находит код JavaScript и сразу же начинает читать его, подготавливая к запуску. Если в коде найдется ошибка, браузер будет стараться продолжить чтение, чтобы пользователю не отказали в показе запрошенной страницы.
4. Браузер начинает выполнять JavaScript-код в момент обнаружения. При этом нужно помнить, что код выполняется вплоть до закрытия страницы, а не до окончания ее формирования.

Код JavaScript может содержаться непосредственно в html-документе либо в отдельном файле с расширением .**js**. Для подключения JavaScript-кода (сценария) к html-документу используется тег **<script>**.

Структура **кода**

В JavaScript код состоит из команд. Сама по себе команда описывает операцию. Набор команд, из которых состоит код, и задает поведение страницы. Команды в коде объявляют переменные, в которых будут храниться значения. В дальнейшем они повлияют на принятие решений.

В переменных могут храниться числа, строки, логические значения и другие данные. Вне зависимости от типа данных все переменные создаются по одному алгоритму.

Чтобы объявить переменную, начинают с ключевого слова **var**. Затем пишут имя переменной. При необходимости можно указать её начальное значение через знак равенства, но это необязательно, так как его можно присвоить позднее.

Оператор **– if**

Для реализации ветвления в JS используется оператор **if**.

Условие — это любое выражение, возвращающее значение *true*, *false*, то есть вопрос, на который ответить можно только «да» или «нет». Если выражение возвращает значение *true*, *false*, то оно будет автоматически к нему приведено. 0, null undefined, “” и NaN будут транслированы в *false*, остальные значения — в *true*. Действие выполняется, когда условие истинно (*true*). Обычно условием является одна или несколько операций сравнения, объединенных логическими связками (И, ИЛИ). В результате проверки условия может выполняться сразу несколько операторов.

В условном операторе (**if**) можно комбинировать условия при помощи логических операций. Например:

if (x || y) — если хотя бы один из аргументов true, то возвращает true, иначе — false.

**Методы** взаимодействия с JavaScript

**Document** метод **querySelector()** возвращает первый элемент (**Element**) документа, который соответствует указанному селектору или группе селекторов в скобках. Если совпадений не найдено, возвращает значение **null**.

Браузерные **события**

**Событие –** это сигнал от браузера о том, что что-то произошло. Самый распространенный пример **события – событие** мыши:

*click* – происходит, когда кликнули на элемент левой кнопкой мыши.

*Использование атрибута HTML*

Обработчик может быть назначен прямо в разметке, в атрибуте, который называется **on**<событие>.

Например, чтобы назначить обработчик события click на элементе input, можно использовать атрибут **onclick.**

**Массивы**

**Массивы** обычно описываются как «объекты, подобные спискам»; они представляют собой в основном отдельные объекты, которые содержат несколько значений, хранящихся в списке. Объекты массива могут храниться в переменных и обрабатываться во многом так же, как и любой другой тип значения, причём разница заключается в том, что мы можем получить доступ к каждому значению внутри списка отдельно и делать полезные и эффективные вещи со списком, а также делать то же самое для каждого из значений.

Массивы в JavaScript являются динамическими: они могут увеличиваться и уменьшаться в размерах по мере необходимости, поэтому можно не указывать фиксированные размеры массивов при их создании или повторном распределении памяти при изменении их размеров. Создать массив проще всего с помощью литерала, который представляет собой простой список разделенных запятыми элементов в квадратных скобках.

*Доступ к элементам массива*

Массивы в JavaScript индексируются с нуля: первый элемент массива имеет индекс, равный 0.

С помощью массива можно выводить различные данные в таблицу. Для начала необходимо определить числа, при совпадении с которыми будет создаваться массив, например:

if (num == '123456' || num == '1234567' || num == '12345678') {

создаем массив:

var massiv =

        [

уровень массива      ['Яблоко', 'Банан', 'Груша', 'Киви', 'Слива],

2-ой уровень массива   ['', '', '', '', '']

        ];

Если число равно 123456, то определим для двухуровневого массива следующие значения, которые будут отображаться в итоговой таблице:

 if (num == '123456') {

                massiv[1][0] = 'Красное';

                massiv [1][1] = 'Желтый';

                massiv [1][2] = 'Спелая';

                massiv [1][3] = 'Дорогое';

                massiv [1][4] = 'Крупная';

                }

Соответственно, если число равно 1234567, то:

if (num == '1234567') {

massiv [1][0] = 'Зеленое';

                massiv [1][1] = 'Черный';

                massiv [1][2] = 'Маленькая';

                massiv [1][3] = 'Фрукт';

                massiv [1][4] = 'Синяя';

                }

Таким образом, число 12345678 запишем аналогично предыдущим:

if (num == '12345678') {

massiv [1][0] = 'Желтое';

                massiv [1][1] = 'Зеленый';

               massiv [1][2] = 'Большая';

                massiv [1][3] = 'Птица';

                massiv [1][4] = 'С косточкой';

                }

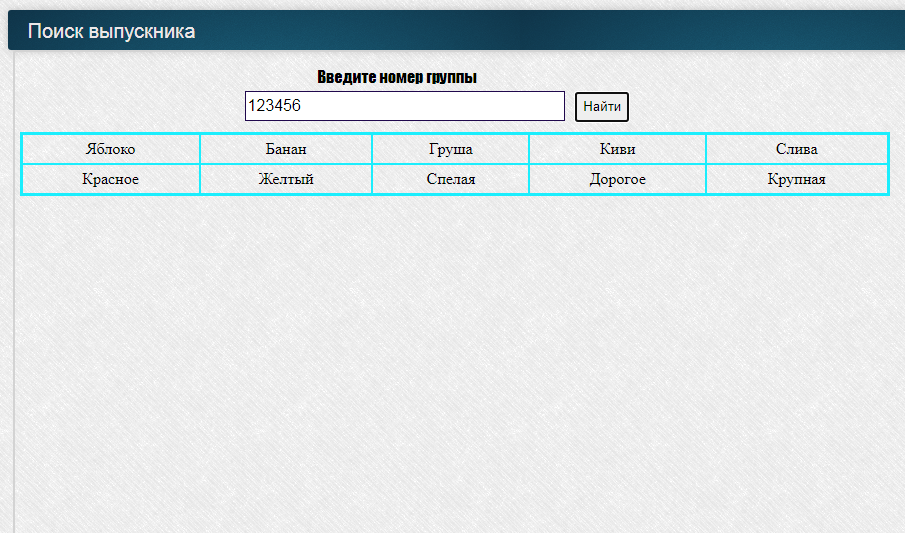
Получаем следующую html-страницу (рисунок 1):

Рисунок 1─ Полученный результат таблицы

**ЗАДАНИЕ**

1. Изучить теоретические аспекты программирования на языке JavaScript.

2. Изучить коды html-документа и описанных для него стилей CSS (см. файл **index.html**, **style.css** в папке с лабораторной работой**)**.

3. В программе **Visual Studio Code** (или Oxygen XML Editor) создать скрипт (расширение файла **.js)**, отображающий массив в виде таблицы.

4. В таблице указать данные о выпускниках учебных групп - 10806199, 10806198, 10806197. Смотреть рекомендации по выполнению скрипта.

**Рекомендации по выполнению скрипта (\*.js)**

1. Для выпускников использовать следующую информацию:

Группа 10806199 – Иван Петрович Новосёлов; 22 года; 9,6;

Группа 10806198 – Василиса Артуровна Шелухина; 21 год; 9,9;

Группа 10806197 – Олег Александрович Нечипоренко; 21 год; 9,5;

2. В качестве исходной страницы, на которую будет выводиться таблица, необходимо использовать шаблон HTML-документа **index.html**, находящийся в папке с данной лабораторной работой.

3. Итоговые таблицы при просмотре с помощью браузера должны выглядеть следующим образом (рис. 2, 3, 4):

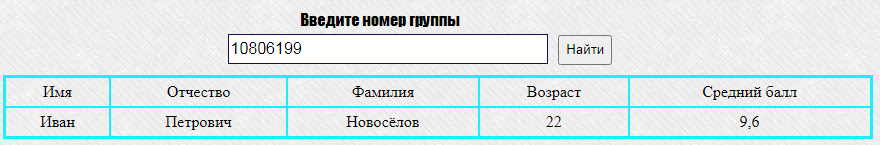


Рисунок 2 ─ Итоговый результат таблицы учебной группы 10806199

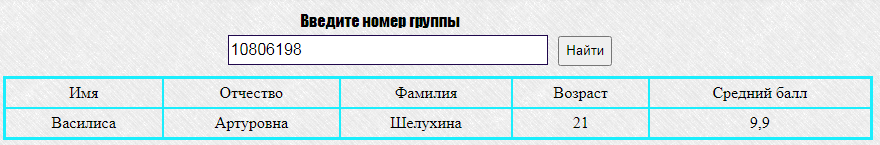


Рисунок 3 ─ Итоговый результат таблицы учебной группы 10806198

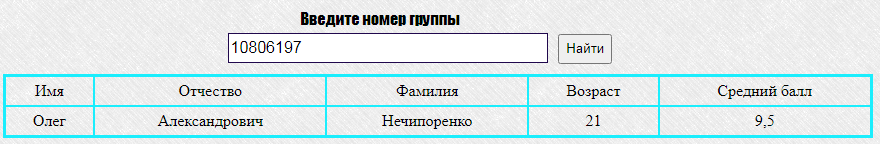


Рисунок 4 ─ Итоговый результат таблицы учебной группы 10806197

5. В документ **index.html** необходимо встроить код для связи с файлом **\*.js**.

6. Результаты продемонстрировать преподавателю и ответить на контрольные вопросы.

**Контрольные вопросы**

1. Как встроить JavaScript-код в html-документ?

2. Для чего используется массивы?

3. С какого значения ведется отсчет индексов массивов в JavaScript?