Министерство образования Республики Беларусь
Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»
Институт информационных технологий

Специальность «Программируемые мобильные системы»

ПРАКТИЧЕСКАЯ РАБОТА №2

По курсу «Языки программирования»

Студент-заочник 2 курса
Группы № 983871
ФИО Щербаков Евгений
Викторович

Минск, 2021

Цель работы: написать Javascript-код для вывода дерева элементов страницы, с которой этот код запущен. Разработать форму для оформления заказа товаров.

Оборудование: ПК, программное обеспечение VS Code, Chrome.

Задание 1

HTML-код:

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>1 задание по JS</title>
    <!-- css -->
    <link rel="stylesheet" href="css/style.css">
    <!-- !css -->
</head>
<body class="body">
    <div>1 уровень вложенности
        <span>2 уровень вложенности
            <span>3 уровень вложенности</span>
        </span>
        <span>2 уровень вложенности
            <span>3 уровень вложенности</span>
            <span>3 уровень вложенности</span>
        </span>
        <span>2 уровень вложенности
            <span>3 уровень вложенности</span>
            <span>3 уровень вложенности</span>
            <span>3 уровень вложенности</span>
        </span>
    </div>

    <div id="rez"></div>
    <!-- js -->
    <script src="js/jq-min.js"></script>
    <script src="js/script.js"></script>
    <!-- !js -->
</body>
</html>
```

CSS-код:

```
* {
  padding-left: 10px;
}

div {
  background-color: grey;
```

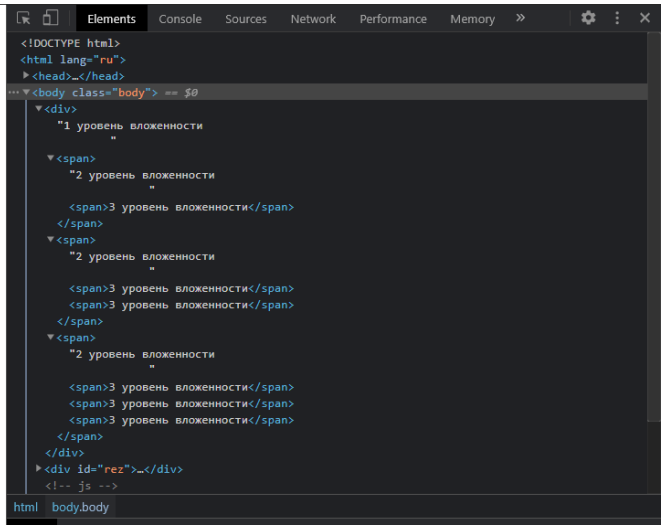
```
}

div:first-of-type * {
  display: -webkit-box;
  display: -webkit-flex;
  display: -ms-flexbox;
  display: flex;
  -webkit-box-orient: vertical;
  -webkit-box-direction: normal;
  -webkit-flex-direction: column;
      -ms-flex-direction: column;
          flex-direction: column;
}
```

JS-код:

```
// TODO: Обход дерева элементов страницы

const tree = (root) => { // Рекурсивная функция для обхода
    let ul = $('<ul></ul>'); // Создаем элемент ul, который используется в роли родителя
    root.children().each(function(index) { // Внутри него перебираем все дочерние элементы, сч
итываем их индекс
        let li = $('<li></li>'); // Создаем элемент li для каждого дочернего
        li.text($(this).prop('tagName')); // Присваиваем ему текст-имя тега дочернего элемента
        li.append(tree($(this))); // Собственно, рекурсия (повторяем всю функцию, но уже в кач
естве корневого используем нынешний li)
        ul.append(li); // Добавляем li в наш ul

        console.log('Тег ' + $(this).prop('tagName') + ', это ' + (index + 1) +
            ' дочерний элемент элемента ' + root.prop('tagName') + '.'); // Отладочный вывод в
 консоль
    });
    root.append(ul); // Добавляем собранный ul в корневой элемент
    return ul;
}

$('#rez').append(tree($('html'))); // Вызываем функцию обхода и собранный список запихиваем в
блок с id="rez"
```

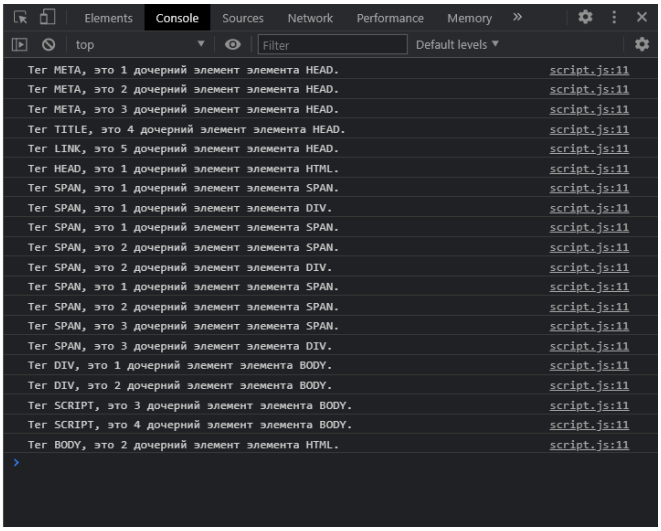
Примеры обхода дерева элементов страницы:

1 уровень вложенности
2 уровень вложенности
3 уровень вложенности
2 уровень вложенности
3 уровень вложенности
3 уровень вложенности
2 уровень вложенности
3 уровень вложенности
3 уровень вложенности
3 уровень вложенности



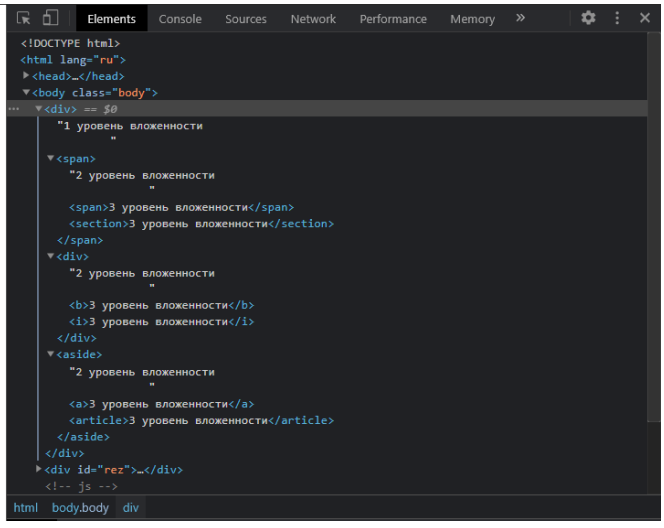
- HEAD
  - META
  - META
  - META
  - TITLE
  - LINK
- BODY
  - DIV
    - SPAN
      - SPAN
    - SPAN
      - SPAN
      - SPAN
    - SPAN
      - SPAN
      - SPAN
      - SPAN
  - DIV
  - SCRIPT
  - SCRIPT

1 уровень вложенности
2 уровень вложенности
3 уровень вложенности
2 уровень вложенности
3 уровень вложенности
3 уровень вложенности
2 уровень вложенности
3 уровень вложенности
3 уровень вложенности
3 уровень вложенности

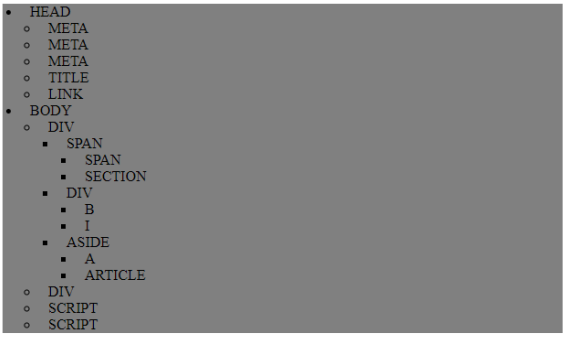


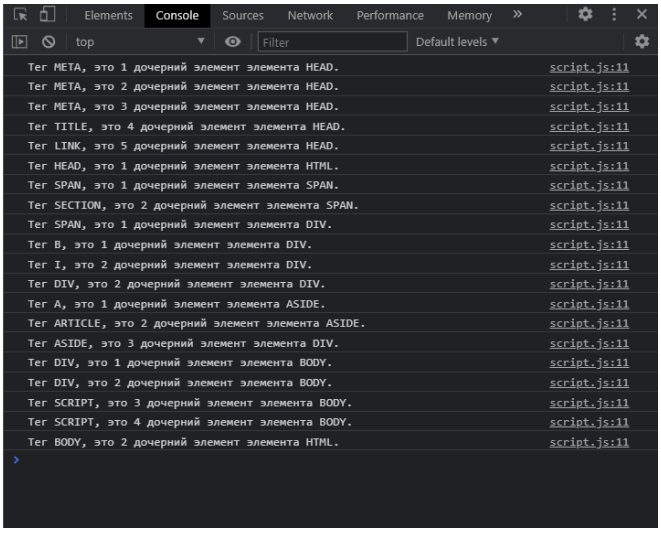
- HEAD
  - META
  - META
  - META
  - TITLE
  - LINK
- BODY
  - DIV
    - SPAN
      - SPAN
    - SPAN
      - SPAN
      - SPAN
    - SPAN
      - SPAN
      - SPAN
      - SPAN
  - DIV
  - SCRIPT
  - SCRIPT

1 уровень вложенности
2 уровень вложенности
3 уровень вложенности
3 уровень вложенности
2 уровень вложенности
3 уровень вложенности
3 уровень вложенности
2 уровень вложенности
3 уровень вложенности
3 уровень вложенности



- HEAD
  - META
  - META
  - META
  - TITLE
  - LINK
- BODY
  - DIV
    - SPAN
      - SPAN
      - SECTION
    - DIV
      - B
      - I
    - ASIDE
      - A
      - ARTICLE
  - DIV
  - SCRIPT
  - SCRIPT





Задание 2

HTML-код:

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>2 задание по JS</title>
    <!-- css -->
    <link rel="stylesheet" href="css/style.css">
    <!-- !css -->
</head>
<body class="body">
    <!-- Форма для ввода данных -->
    <form action="">
        <div>
            <label for="name">Наименование товара:</label>
            <input type="text" name="name" id="name" required>
        </div>
        <div>
            <label for="quantity">Количество:</label>
            <input type="number" name="quantity" id="quantity" required>
            <span>шт.</span>
        </div>
        <div>
            <label for="cost">Стоимость 1 шт:</label>
            <input type="number" name="cost" id="cost" required>
            <span>руб.</span>
        </div>

        <div>
            <button id="reset">Очистить таблицу</button>
            <button id="calc">Добавить в заказ</button>
        </div>
    </form>
```

```
    <!-- Форма для ввода данных -->

    <!-- Блок с таблицей -->
    <div id="rez"></div>
    <!-- Блок с таблицей -->

    <!-- js -->
    <script src="js/jq-min.js"></script>
    <script src="js/script.js"></script>
    <!-- !js -->
</body>
</html>
```

CSS-код:

```
body,
form {
  display: -webkit-box;
  display: -webkit-flex;
  display: -ms-flexbox;
  display: flex;
  -webkit-box-orient: vertical;
  -webkit-box-direction: normal;
  -webkit-flex-direction: column;
      -ms-flex-direction: column;
          flex-direction: column;
  -webkit-box-align: center;
  -webkit-align-items: center;
      -ms-flex-align: center;
          align-items: center;
}

* {
  margin: 0;
  padding: 0;
}

form {
  background-color: darkgrey;
}

form div {
  width: 600px;
  height: 50px;
  display: -webkit-box;
  display: -webkit-flex;
  display: -ms-flexbox;
  display: flex;
  -webkit-box-align: center;
  -webkit-align-items: center;
      -ms-flex-align: center;
          align-items: center;
}

form div label {
  padding: 0 10px;
  -webkit-flex-basis: 30%;
```

```
      -ms-flex-preferred-size: 30%;
          flex-basis: 30%;
}

form div input[type="text"],
form div input[type="number"] {
  height: 30px;
  -webkit-flex-basis: 50%;
      -ms-flex-preferred-size: 50%;
          flex-basis: 50%;
}

form div span {
  padding: 10px;
}

form div button {
  margin: 0 auto;
  width: 240px;
  height: 30px;
  background-color: #888;
  border: 2px solid white;
  border-radius: 15px;
  cursor: pointer;
}

form div button:hover {
  color: green;
}

table {
  width: 600px;
  background-color: #aaa;
}

table caption {
  padding: 10px;
  color: darkslateblue;
  text-transform: capitalize;
  font-weight: bold;
}

table td {
  padding: 5px;
  width: 25%;
  text-align: center;
}

table th {
  padding: 10px;
}

table tr:nth-child(odd) {
  background-color: lightblue;
}

table tr:nth-child(even) {
  background-color: lightgreen;
}
```

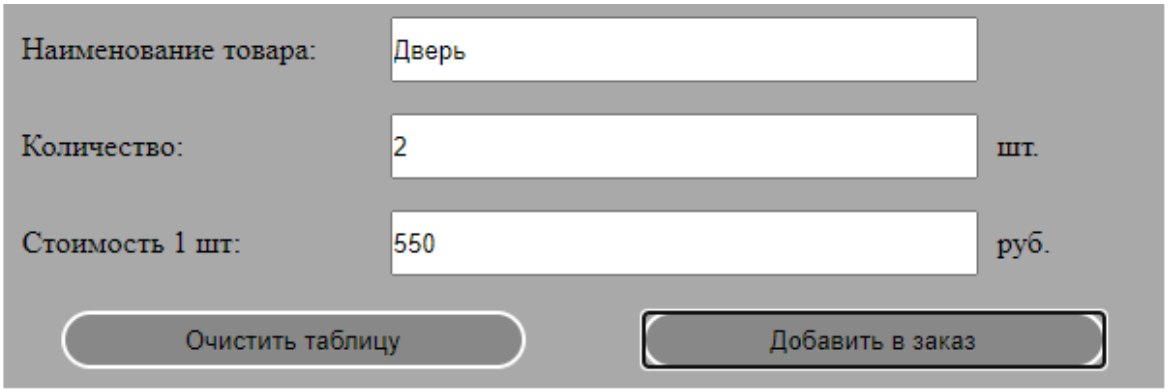
```
#summary td {
  color: darkred;
  font-weight: bold;
}
```

JS-код:

```
// TODO: ФУНКЦИИ

const addRow = () => { // Функция для добавления строки в таблицу
    let row = $('<tr></tr>'); // Создаем экземпляр строки
    row.append($('<td></td>').text($('#name').val())); // Запихиваем туда ячейку с названием т
овара
    row.append($('<td></td>').text($('#quantity').val())); // Запихиваем туда ячейку с количес
твом
    row.append($('<td></td>').text($('#cost').val())); // Запихиваем туда ячейку с ценой 1 шту
ки
    row.append($('<td></td>').text($('#quantity').val() * $('#cost').val())); // 4 ячейка с ра
считанной стоимостью
    return row; // Собранная строчка
}

const tdHover = () => { // Обработчик положения мыши для ячеек со стоимостью
    $('#rez tr:not(:first-
type)').each(function() { // Перебрать все строки таблицы, кроме первой
        $(this).find('td:last-
child').on('mouseover', function() { // Если мышь над ячейкой стоимости
            $(this).css('background', 'lightyellow'); // Изменить цвет фона
        });
        $(this).find('td:last-
child').on('mouseout', function() { // Если мышь не над ячейкой стоимости
            $(this).css('background', 'transparent'); // Убрать цвет фона
        });
    });
}

// TODO: ОБРАБОТЧИКИ ДЛЯ КНОПОК В ФОРМЕ

$('#reset').on('click', function(e) { // Обработчик нажатия на кнопку сброса
    e.preventDefault(); // Останавливаем переход по ссылке
    $('#rez').empty(); // Очищаем блок с таблицей
});

$('#calc').on('click', function(e) { // Обработчик нажатия на кнопку расчета стоимости
    e.preventDefault(); // Останавливаем переход по ссылке
    let sum = 0; // Счетчик для подсчета итоговой суммы

    if ($('#name').val() != '' && $('#quantity').val() != '' && $('#cost').val() != '') { // П
роверка на пустоту полей формы
        if ($('#rez').html() == '') { // Если таблица еще не создана
            let table = $('<table></table>'); // Создаем экземпляр таблицы
            table.append($('<caption>Заказ</caption>')); // Впихиваем в него описание
            table.append($('<tr><th>Название</th><th>Количество</th><th>Стоимость 1 шт., руб.<
/th>><th>Суммарная стоимость, руб.</th></tr>'));
                // Впихиваем в него заглавную строку
            table.append(addRow()); // Впихиваем строку с нынешними параметрами формы
```

```
            $('#rez').append(table); // Впихиваем полчившуюся таблицу в блок с результатом
        } else { // Если таблица уже существует
            $('#summary').remove(); // Удаляем строку с итоговой стоимостью
            $('#rez table').append(addRow()); // Впихиваем строку с нынешними параметрами форм
ы
        }

        $('#rez tr:not(:first-of-type)').each(function() { // Перебираем все ячейки 4 столбика
            sum += Number($(this).find('td:last-
child').text()); // Складываем их значения, дабы найти итоговую стоимость
        });

        $('#rez table').append($('<tr id="summary"><td colspan="3">Сумма заказа:</td><td>' + s
um + '</td></tr>'));
            // Добавляем в таблицу строчку с итоговой стоимостью заказа
        tdHover();
    } else alert('СИНИЙ ЭКРАН: не введены обязательные параметры'); // Если поля пустые - ошиб
ка
});
```

Пример выполнения программы:



Заказ

Название	Количество	Стоимость 1 шт., руб.	Суммарная стоимость, руб.
Комп	1	2500	2500
Холодильник	1	600	600
Дверь	2	550	1100
Сумма заказа:			**4200**

Вывод: в ходе работы написан Javascript-код для вывода дерева элементов страницы, с которой этот код запущен. Разработана страница для оформления заказа товаров.