ПРАКТИЧЕСКАЯ РАБОТА №1

Создание структуры проекта. Выбор текстового редактора. Базовая разметка страницы

Создание структуры проекта

Создание любого проекта начинается с создания базовой структуры папок.

Эта структура является однотипной:

В удобном для вас месте создайте папку с вашим будущем проектом. В корне этой папке создайте папку “static” (здесь будут хранится статические файлы: изображения, стили) и документ “index.html” (это базовый файл, с которым нам доведётся работать). В папке “static” создайте папку “css” и “images”. Последним штрихом добавляем файл “style.css” в папку “css”.

Примечание: Кроме того, более динамические сайты используют для интерактивных элементов, которые создаются с помощью JavaScript. И файлы со скриптами хранятся еще в папке “js”, которая находится в папке “static”.

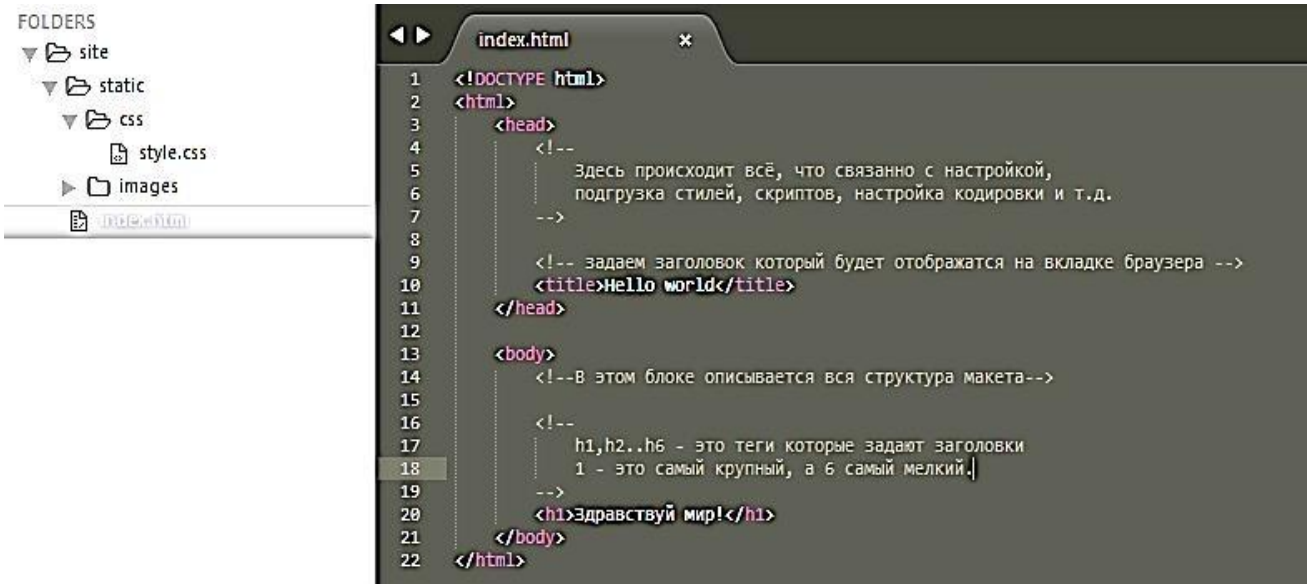
Выбор текстового редактора

HTML - страницу можно сверстать, используя Блокнот или Notepad++, но на данный момент существуют редакторы кода, имеющие огромный функционал в

виде подсветки синтаксиса, открытие целых директорий, множества файлов, разбитие на несколько окон и т.д. Одни из самых популярных это SublimeText, Brackets, Atom. Весь код, представленный в дальнейшем в описании лабораторных работ будет оформлен в SublimeText3.

Базовая разметка страницы

Откройте файл “index.html” и перепишите вручную текст с картинки. Всё, кроме того что заключено в теги вида <!-- какой-то комментарий -->(текст в данных тегах не будет отображаться в веб браузере). И посмотрите, что в браузере получилось.



Пример кода

В результате вы написали базовое приложение типа “Hello world”. Самым главным, что стоит запомнить из этого примера, что теги - это элементы языка вида <имя тега> и они обязательно должны закрываться: </имя тега>. В дальнейшем содержимое тегов может форматироваться. Указанная структура документа является базовой и обязательной.

index.html

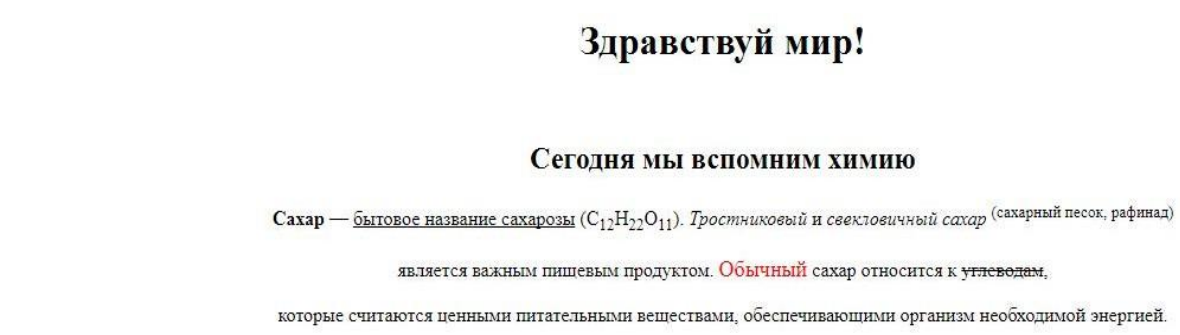
```
<!DOCTYPE html>
<html>
    <head>
        <title>Hello world</title>
        <!--Для того, чтобы браузеры понимали кирилические символы введите:-->
        <meta charset="utf-8">
    </head>
    <!--
                    Примечание
    теги можно писать как полностью большими так и полностью маленькими буквами,
    разницы никакой не будет. Тоесть <H1> тоже самаое, что и <h1>
    -->
    <body>
        <!--div - это обычный контейнер который группирует элементы в единое целое -->
        <!-- к div-ам мы еще еще более подробнее в дальнейшем -->
        <div align="center">
            <h1>Здравствуй мир!</h1>

            <!-- -->
            <br>

            <h2>Сегодня мы вспомним химию</h2>
            <!--разбивает текст на параграфы -->
            <p>
                <b>Сахар</b> – <u>бытовое название сахарозы</u> (C<sub>12</sub>H<sub>22</sub>O<sub>11</sub>).
                <i>Тростниковый</i> и <i>свекловичный сахар</i> <sup>(сахарный песок, рафинад)</sup>

                <!--
                    B - задает полужирный шрифт
                    U - задает нижнее подчеркивание
                    I - курсивный шрифт
                    SUB - подстрочный индекс
                    SUP - подстрочный индекс
                -->
            </p>

            <p>
                является важным пищевым продуктом. <font color="red" size="4">Обычный</font> сахар относится к
                <strike>углеводам</strike>,
```

Работа с различными тэгами

```
                <!--
                    strike - зачеркивание
                    font - Тег для задание свойств шрифта

                    На практике задание свойств текста через теги задается с помощью CSS,
                    но для базового ознакомления будет полезным ознакомится с этим. В следующей
                    лабораторной работе мы проделаем те же самые вещи с помощью CSS
                -->
                <!-- BR - перевод строки -->
                <!--
                    Вы уже обратили, что водном из тегов была такая вещь, как align="center",
                    это мы назначали тегу напрямую стиль, который задает центральное позиционирование,
                    всему что находится в этом теге, кроме того, можно задать позиционирование: left(слева), right(
                    справа), top(сверху), bottom(снизу)
                -->

                <!--
                    Font- это тег, позволяющий форматировать нам текст, на этом примере мы задали цвет через
                    свойство "color" и размер шрифта "size", кроме того цвет можно задавать не только
                    зарезервированным словом, а его кодом, к примеру #000000  задаст черный цвет, подобрать код
                    цвета через палитру вы можете на сайте "getcolor.ru"
                -->
            </p>

            <p>
                которые считаются ценными питательными веществами, обеспечивающими организм необходимой энергией.
            </p>
        </div>
    </body>
</html>
```

Работа с различными тэгами

В комментариях вы можете увидеть описание каждого нового тега. Открывая страницув браузере, вы получите такой результат:



Объявление стилей. Базовые свойства стилей стилей.

Рассмотрим форматирование текста в HTML-документе с помощью таблицы каскадных стилей (далее CSS - CascalSheetStyles).
Существуют различные способы объявление стилей: в самих тегах, в теге <style></style>, и в отдельном файле. Мы же рассмотрим последние два метода.

Для начала создайте новый документ «index03.html» и задайте ему базовую структуру,как в лабораторной работе № 1 и новое содержимое блока <style> как в примере:

index.html × index03.html ×

```
<!DOCTYPE html>
<html>
<head>
    <title>add style</title>
    <style type="text/css">
        h2{
            color: red;
        }
    </style>
    <!--
        Что мы сделали? Сы сказали нашей странице, что всем h2 присвоить крассный цвет.
        Больше об оформлении CSS мы поговорим далее.
        А пока стоит усвоить такой принцип задание свойств:

        то-чему-зажается-свойство{
            свойство: занчение;
            ...
            свойство: значение;
        }
    -->
</head>
<body>
    <h2>Давайте что-то новенькое напишем</h2>
</body>
</html>
```

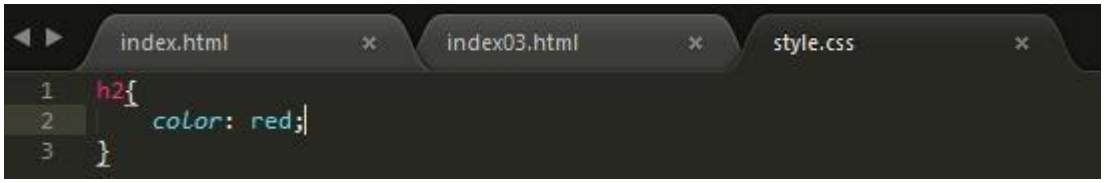
Объявление стилей

Подключение внешнего стиля

Для этого вместо блока <style> вставьте эту строчку:

```
4       <link rel="stylesheet" type="text/css" href="static/css/style.css">
5       <!--На этом этапе мы указываем путь к нашему будующему файлу стилей,
6       который находится в static -> css -> style.css-->
```

Затем открываем файл “style.css” из “static/css” и добавляем следующий код:



Изменение цвета заголовка

Обновив страницу, вы обнаружите, что заголовок так и останется красным. Так и должно быть, так как вы логику описания стилей перенесли в отдельный файл.

Как работают классы

Для того чтобы тегу присвоить класс в самом теге вам надо написать class=”именна_классов ” (имена должны писаться латинскими символами и использовать из сторонних символов только дефис и нижнее подчеркивание).

В самом же файле стилей, стилю задаются свойства таким образом

.имя_класса {свойства}.

В теге объявляется через id=”имя_id” в стилях #имя_id{свойства}.

id это уникальное имя, которое следует объявлять только лишь раз один раз на странице, а класс сколько угодно раз.

Повторите следующий код и посмотрите результат:

index03.html

```
<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" type="text/css" href="static
    /css/style.css">
</head>
<body>
    <h2 class="my-title1">Давайте что-то новенькое
    напишем</h2>
    <h2 class="my-title2">Еще текст?</h2>
    <h5 class="my-title1">Текста много не бывает</h5>
    <p id="green">Один и тот же id</p>
</body>
</html>
```

style.css

```
body{
    margin: 0px;
    padding: 0px;
}
.my-title1{
    color: red;
}
.my-title2{
    color: orange;
}
#green{
    color: green
}
```

ПРАКТИЧЕСКАЯ РАБОТА №2

Списки. Типы списков. Ссылки. Hover-эффекты.

Для отображения различных элементов в связанной структуре используют такой элемент, как списки. На следующем рисунке представлен код по созданию простой шапки сайтас использованием немаркированных списков.

Прежде чем повторять код, примите к сведению, что у элементов <li> (элемент самого списка), есть html-атрибут «type». Этот атрибут может принимать различные значения, например «disc» (элементы списка будут иметь круглый черный маркер), «circle» (маркеры будут в виде кольца) и «square»(квадратные маркеры).

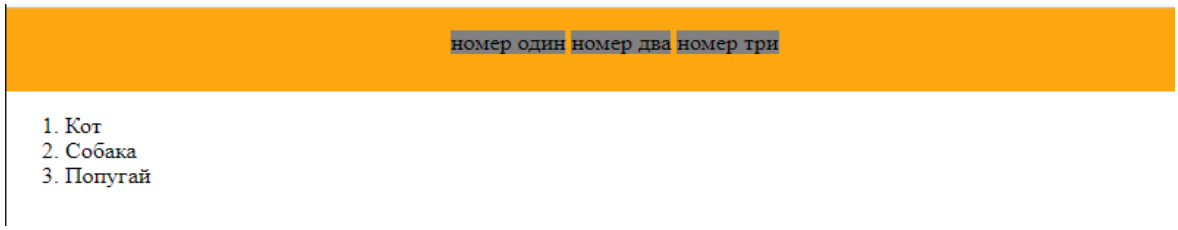
index04.html

```html
<!DOCTYPE html>
<html>
<head>
    <title>lab-04</title>
    <link rel="stylesheet" type="text/css" href="static/css/
    style.css">
</head>
<body>
    <!--nav - наш тег для шапки сайта-->
    <nav>
        <ul class="list">
            <!--ul - контейнер элементов списка -->
            <!--li - эллемент списка -->
            <li>номер один</li>
            <li>номер два</li>
            <li>номер три</li>
        </ul>
    </nav>

    <!--Блок с контентом-->
    <content>
        <!-- ol - контейнер нумерованого списка-->
        <ol>
            <li>Кот</li>
            <li>Собака</li>
            <li>Попугай</li>
        </ol>
    </content>
</body>
</html>
```

style.css

```css
body{
    margin: 0px;
    padding: 0px;
}
nav{
    height: 60px;
    width: 100%;
    background-color: #FEA610;/*оранжеаый фон*/
}
/*задаем стили контейнеру с классом list*/
ul.list{
    position: absolute;
    margin-left: calc((100% - 800px)/2);
    width: 800px;
    height: 30px;
    list-style-type: none;/*убираем маркировку li-элементов*/

}

ul.list li{
    background-color: grey;/*фон элементов серый*/
    display: inline;/*li - элементы располагаются в ряд*/
}
```

В результате вы будете иметь такой результат:



1. Кот
2. Собака
3. Попугай

Давайте теперь создадим на элементах списка ссылки и добавим к ней простую анимацию.

Ссылки описывает тег <a>.

Основной атрибут этого тега является «href», href – это

ссылка на страницу.

В ссылке может быть любой объект (тег): текст, изображение, блок и т.д.

Стандартно HTML задает тексту в ссылке подчеркивание и голубой цвет шрифта и изменение цвета шрифта ссылки на фиолетовый, после перехода по этой ссылке. Но с помощью css или тегов html атрибутов можно изменить эти стандартные свойства.

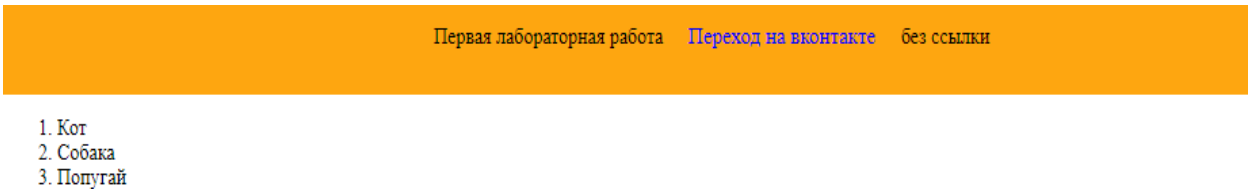
Кроме того, в CSS есть такое понятие, как псевдокласс. Псевдокласс в CSS - это ключевое слово, добавленное к селектору, которое определяет его особое состояние.

Например, :hover применит стиль, когда пользователь наводит курсор на элемент, указанныйселектором. Как раз этот псевдокласс и применим к ссылке и посмотрим, что получится.

```html
<!DOCTYPE html>
<html>
<head>
    <title>lab-04</title>
    <link rel="stylesheet" type="text/css" href="static/css/
    style.css">
</head>
<body>

    <nav>
        <ul class="list">
            <a href="index.html">
                <li>Первая лабораторная работа</li>
            </a>
            <a href="https://vk.com">
                <li>Переход на вконтакте</li>
            </a>
            <li>без ссылки</li>
        </ul>
    </nav>

    <content>
        <ol>
            <li>Кот</li>
            <li>Собака</li>
            <li>Попугай</li>
        </ol>
    </content>

</body>
</html>
```

```css
body{
    margin: 0px;
    padding: 0px;
    font-size: 14px;/*задаем размер шрифта для всего документа*/
}
nav{
    height: 60px;
    width: 100%;
    background-color: #FEA610;
}
ul.list{
    position: absolute;
    margin-left: calc((100% - 800px)/2);
    width: 800px;
    height: 30px;
    list-style-type: none;

}
ul.list li{
    margin-left: 15px;
    display: inline;
}
a{
    color: black;
    text-decoration: none;/*отменяем нижнее подчеркивание*/
}
a:hover{
    color: blue;
    font-size: 15px
}
```

Ниже представлен результат написанного вами кода:



Пример совсем не совершенен, но позволяет опробовать новые изученные свойства.

ПРАКТИЧЕСКАЯ РАБОТА №3

Блоки. Изображения. Позиционирование. Отступы.

Блок (атрибут), это сущность, которая напоминает контейнер (коробку), которая хранит в себе различные элементы с другими свойствами (размер, цвет и т.д.). Вложенные элементы в этот контейнер наследуют многие из свойств (такие, как шрифт, цвет, текстовое форматирование). В CSS есть свойства, которые применяются в основном к блоку и отвечают за способы расположения блоков.

Следующий пример покажет, как создать плитку цветных блоков, которой мы воспользуемся в дальнейшем.

index05.html

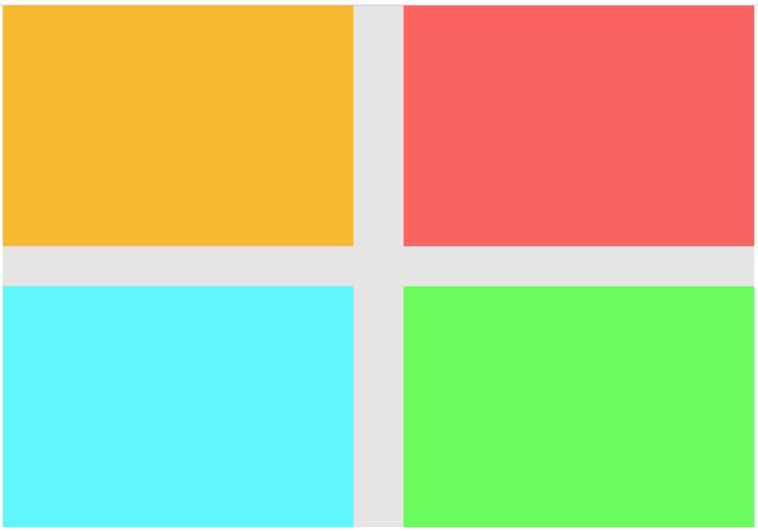
```
<!DOCTYPE html>
<html>
<head>
    <title>lab-5</title>
    <link rel="stylesheet" type="text/css" href="static/css/style.css">
</head>
<body>
    <div class="centr-cont">
        <div class="block first"></div>
        <div class="block second"></div>
        <div class="block third"></div>
        <div class="block fourth"></div>
    </div>
</body>
</html>
```

style.css

```
body{
    margin: 0px;
    padding: 0px;
    font-size: 14px;/*задаем размер шрифта для всего документа*/
}
.centr-cont{
    height: 650px;
    width: 750px;
    background-color: #E7E4E4;
    margin-left: calc((100% - 750px)/2);
    /*операция калс выполняет мат операции,
    очень важно между - и + с двух сторон ставить пробелы*/
}
.block{
    position: relative;
    height: 300px;
    width: 350px;
    display: inline-block;
    /*элементыэтого класса будут следовать в линии друг за другом,
    пока им позволяеет ширина окна браузера*/
}
.first{
    background-color: #FAB933;
}
.second{
    background-color: #FC625F;
    margin-left: 46px;
}
.third{
    background-color: #5FF6FC;
    margin-top: 46px;
}
.fourth{
    margin-left: 46px;
    margin-top: 46px;
    background-color: #69FC5F;
}
```

Рисунок 14 – Блоки

Результат:



Теория

```
/*
    Свойство padding позволяет задать величину поля
сразу для всех сторон элемента или определить ее только для указанных сторон.
Применяется для текста. И эти отступи применятся относительно того блока,
в котором он находится.

        Пример:
    padding-left: 25px;
    padding: 10px 30px;
    padding: 20px;

    Свойство margin Устанавливает величину отступа от каждого края элемента.
Отступом является пространство от границы текущего элемента до внутренней
границы его родительского элемента.

        Пример:
    padding-left: 25px;
    padding: 10px 30px;
    padding: 20px;

    Кроме того отступы можно задавать не только в пикселях(px),
но и процентах(%).
*/
```

Позиционирование (css свойство- position)

Position - устанавливает способ позиционирования элемента относительно окна браузера или других объектов на веб-странице.

Способ обьявления: position: absolute | fixed | relative | static | inherit

absolute

Указывает, что элемент абсолютно позиционирован, при этом другие элементы отображаются на веб-странице словно абсолютно позиционированного элемента и

нет. Положение элемента задается свойствами left, top, right и bottom, также на положение влияет значение свойства position родительского элемента. Так, если у родителя значение position установлено как static или родителя нет, то отсчет координат ведется от края окна браузера.

Если у родителя значение position задано как fixed, relative или absolute, то отсчет координат ведется от края родительского элемента.

fixed

По своему действию это значение близко к absolute, но в отличие от него привязывается к указанной свойствами left, top, right и bottom точке на экране и не меняет своего положения при прокрутке веб-страницы. Браузер Firefox вообще не отображает полосы прокрутки, если положение элемента задано фиксированным, и оно не помещается целиком в окно брау- зера. В браузере Opera хотя и показываются полосы прокрутки, но они никак не влияют на позицию элемента.

relative

Положение элемента устанавливается относительно его исходного места. Добавление свойств left, top, right и bottom изменяет позицию элемента и сдвигает его в ту или иную сторону от первоначального расположения.

static

Элементы отображаются как обычно. Использование свойств left, top, right и bottom неприводит к каким-либо результатам.

inherit

Наследует значение родителя.

Чтобы сделать еще красивее, добавим несколько изображений. Изображения очень похожи на ссылки, но имеют некоторые различия. Тег - <img> (не закрывающийся), атрибут определяющий путь к файлу – src=”путь”. Путь указывается также, как и у ссылки, только является важным разрешение файла (например, .jpg)

Скачаем 4 изображения из интернета. Можно воспользоваться яндекс картинками. Скачанные изображения поместим в раннее созданную папку «images», а для

большей простоты изображения переименуем в по образцу «01.расширение_изображения».

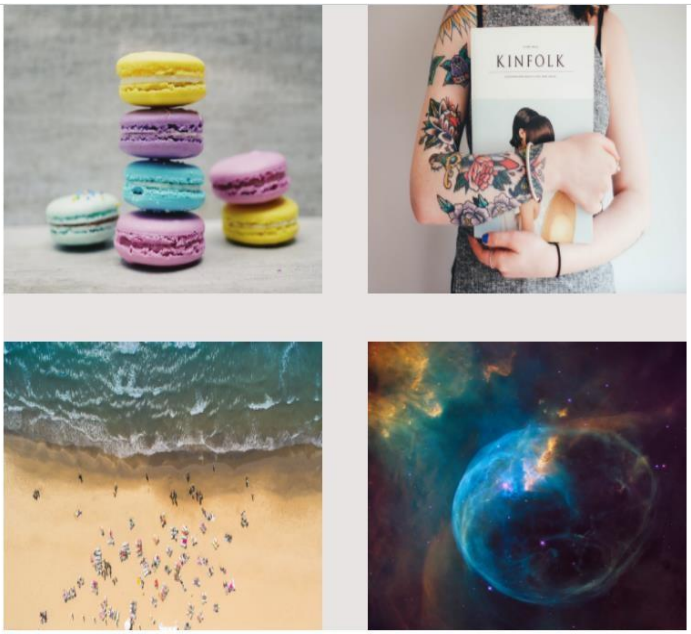
index05.html

```
<!DOCTYPE html>
<html>
<head>
    <title>lab-5</title>
    <link rel="stylesheet" type="text/css" href="static
    /css/style.css">
</head>
<body>
    <div class="centr-cont">
        <div class="block first">
            <img src="static/images/01.jpg">
        </div>
        <div class="block second">
            <img src="static/images/02.jpg">
        </div>
        <div class="block third">
            <img src="static/images/03.jpg">
        </div>
        <div class="block fourth">
            <img src="static/images/04.jpg">
        </div>
    </div>
</body>
</html>
```

style.css

```
body{
    margin: 0px;
    padding: 0px;
}
img{
    height: 300px;
    width: 350px;
}
.centr-cont{
    height: 650px;
    width: 750px;
    background-color: #E7E4E4;
    margin-left: calc((100% - 750px)/2);
}
.block{
    position: relative;
    height: 300px;
    width: 350px;
    display: inline-block;
}
.second{
    margin-left: 46px;
}
.third{
    margin-top: 46px;
}
.fourth{
    margin-left: 46px;
    margin-top: 46px;
}
```

Результат:



Существует способ задания фона блоку через css-свойство «background-img», для ознакомления рекомендуем изучить его самостоятельно и опробовать на одном из блоков.

ПРАКТИЧЕСКАЯ РАБОТА №4

Таблицы.

Прежде чем приступить к выполнению лабораторной создайте файл «index0?.html» и файл стилей для него «style0?.css»

Создание простейшей таблицы

Создание таблицы начинается с тега **<table></table>**, далее, внутри него располагаетсявсе содержимое, а именно строки **<tr>** и уже внутри строк ячейки **<td>.** Чтобы понять, как всеэто работает обратите внимание на пример ниже:

Примечание: заголовочную ячейку можно создать тегом <th>, текст в такой ячейке располагается посередине и выделяется жирным шрифтом.

Для начала зададим базовые стили

```
1  /* Задаем рамку таблице */
2  table{
3      border:1px solid black;
4  }
5
6  /* Задаем рамку ячейке */
7  td{
8      border:1px solid black;
9  }
```

Рисунок 18 – Базовый стиль

не забудьте подключить файл стилей к вашему HTML файлу.

Создадим простейшую таблицу и посмотрим на результат в браузере

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="style07.css">
</head>
<body>
    <table>
        <tr>
            <td>Университет</td>
            <td>Кол-во студентов</td>
        </tr>
        <tr>
            <td>ДГТУ</td>
            <td>46 000</td>
        </tr>
        <tr>
            <td>ЮФУ</td>
            <td>33125</td>
        </tr>
        <tr>
            <td>РИНХ</td>
            <td>21 636</td>
        </tr>
    </table>
</body>
</html>
```

Рисунок 19 - Создание таблицы

Университет	Кол-во студентов
ДГТУ	46 000
ЮФУ	33125
РИНХ	21 636

Рисунок 20 - Результат

Задание 1

Дополнить таблицу как показано ниже:

Университет	Кол-во студентов	Год основания	Адрес
ДГТУ	46 000	1930	пл.Гагарина,1
ЮФУ	33125	2006	ул.Б.Садовая,105
РИНХ	21 636	1931	ул.Б.Садовая,69

Самостоятельно освойте свойство **border-collapse**, в выполнении задания оно обязательно понадобится.

2. Работа с рамками, отступами внутри и между ячейками, ячейки-заголовки

Иногда бывают ситуации, когда нужно чтобы отображалась только нижняя рамка ячейки и т.д, такие эффекты достигаются за счет свойств, представленных ниже:

border-right, border-left, border-top, border-bottom

Примечание: для того чтобы убрать рамку в значении свойства напишите **none**

Задание 2

В созданной таблице из Задания 1 оставьте рамки только справа и слева как на примере
ниже:

Университет	Кол-во студентов	Год основания	Адрес
ДГТУ	46 000	1930	пл.Гагарина,1
ЮФУ	33125	2006	ул.Б.Садовая,105
РИНХ	21 636	1931	ул.Б.Садовая,69

Отступы внутри ячеек задаются с помощью свойства **padding:**

padding: 5px - одно значение задает отступ всем сторонам,
padding: 5px 10px - первое значение задает отступ сверху-снизу, второе справа-слева.
padding: 5px 10px 5px - первое значение задает отступ сверху, второе одновременно справа-слева, третье снизу.
padding: 5px 10px 5px 10px - первое сверху, второе справа, третье снизу, четвертое слева.

Задание 3

Задайте ячейкам отступы сверху-снизу (5px), справа-слева (10px) как на примере ниже:

Университет	Кол-во студентов	Год основания	Адрес
ДГТУ	46 000	1930	пл.Гагарина,1
ЮФУ	33125	2006	ул.Б.Садовая,105
РИНХ	21 636	1931	ул.Б.Садовая,69

Задание 4

Замените тег **<td>** в категориях на **<th>.**
В стилях задайте тегу <th> рамку и внутренний отступ (10px)

Университет	Кол-во студентов	Год основания	Адрес
ДГТУ	46 000	1930	пл.Гагарина,1
ЮФУ	33125	2006	ул.Б.Садовая,105
РИНХ	21 636	1931	ул.Б.Садовая,69

Объединение ячеек в строках-столбцах

Для объединения двух и более ячеек используют атрибуты colspan и rowspan для тега

<td>

colspan — устанавливает число объединяемых ячеек по горизонтали rowspan — аналогично, только по вертикали

Следуя инструкциям, создайте таблицу:

Браузер	Посещения	
	Количество	В процентах
Mozila Firefox	163	59%
Google Chrome	78	28%
Safari	35	13%

Задайте базовую структуру таблицы и стили

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="style07.css">
</head>
<body>
    <table>
        <tr>
            <th>Браузер</th>
            <th>Посещения</th>
        </tr>
        <tr>
            <th>Количество</th>
            <th>В процентах</th>
        </tr>
        <tr>
            <td>Mozila Firefox</td>
            <td>163</td>
            <td>59%</td>
        </tr>
        <tr>
            <td>Google Chrome</td>
            <td>78</td>
            <td>28%</td>
        </tr>
        <tr>
            <td>Safari</td>
            <td>35</td>
            <td>13%</td>
        </tr>
    </table>
</body>
</html>
```

```css
table{
    border:1px solid black;
    border-collapse: collapse;
}

td{
    border:1px solid black;
    padding: 10px;
}

th{
    border:1px solid black;
    padding:10px;
}
```

Начинаем работать с объединением ячеек

Первый тег <th> занимает две ячейки по горизонтали и две ячейки по вертикали, поэтому задаем:

<th colspan = “2“ rowspan = “2“></th>

Второй занимает две ячейки по горизонтали:

<th colspan = “2“></th>

Тег <th> с названиями браузеров точно так же как и первый занимает две ячейки по горизонтали:

<th colspan = “2“></th>

Проделав все действия выше, вы должны получить такой код и результат изображенный выше:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="style07.css">
</head>
<body>
    <table>
        <tr>
            <th colspan="2" rowspan="2">Браузер</th>
            <th colspan="2">Посещения</th>
        </tr>
        <tr>
            <th>Количество</th>
            <th>В процентах</th>
        </tr>
        <tr>
            <td colspan="2">Mozila Firefox</td>
            <td>163</td>
            <td>59%</td>
        </tr>
        <tr>
            <td colspan="2">Google Chrome</td>
            <td>78</td>
            <td>28%</td>
        </tr>
        <tr>
            <td colspan="2">Safari</td>
            <td>35</td>
            <td>13%</td>
        </tr>
    </table>
</body>
</html>
```

Задание 5

Создайте таблицу, изображенную ниже (размеры отступов кратны 5):

<table>
<tr><th>Город</th><th>Посещения</th><th>Страниц</th><th>Время</th></tr>
<tr><td>СПб</td><td>199</td><td>18,02</td><td>00:13:45</td></tr>
<tr><td>Москва</td><td>69</td><td rowspan="2">нет данных</td><td>00:00:44</td></tr>
<tr><td>Киев</td><td>5</td><td>00:18:07</td></tr>
<tr><td colspan="3">Всего посещений</td><td>273</td></tr>
</table>

ПРАКТИЧЕСКАЯ РАБОТА №5

Формы

Тег **<form>** устанавливает форму на веб-странице. Форма предназначена для обмена данными между пользователем и сервером. Область применения форм не ограничена отправкой данных на сервер, с помощью клиентских скриптов можно получить доступ к любому элементу формы, изменять его и применять по своему усмотрению.

Для отправки формы на сервер используется кнопка Submit, того же можно добиться, если нажать клавишу Enter в пределах формы. Если кнопка Submit отсутствует в форме, клавиша Enter имитирует ее использование.

Атрибут action в форме принимает ссылку(URL) который ведет либо на локальный скрипт, либо на сервер которые примут на себя обработку данных формы.

Поля в форме именуются тегом <input> имеют один важный атрибут – type, который определяет тип поля, который будет использован. Type может принимать значения:

Тип	Описание	Вид
button	Кнопка.	Кнопка
checkbox	Флажки. Позволяют выбрать более одного варианта из предложенных.	☐ Пиво ☐ Чай ☐ Кофе
file	Поле для ввода имени файла, который пересылается на сервер.	Выберите файл Файл не выбран
hidden	Скрытое поле. Оно никак не отображается на веб-странице.	
image	Поле с изображением. При нажатии на рисунок данные формы отправляются на сервер.	Отправить
password	Обычное текстовое поле, но отличается от него тем, что все символы показываются звездочками. Предназначено для того, чтобы никто не подглядел вводимый пароль.	
radio	Переключатели. Используются, когда следует выбрать один вариант из нескольких предложенных.	○ Пиво ○ Чай ○ Кофе
reset	Кнопка для возвращения данных формы в первоначальное значение.	Сбросить
submit	Кнопка для отправки данных формы на сервер.	Отправить
text	Текстовое поле. Предназначено для ввода символов с помощью клавиатуры.	

Рисунок 21 - Работа с формами

Чтобы понять данный материал выполните следующий код (так как у нас нет сервера,то никаких действий не выполнится после подтверждения формы):

```
<!DOCTYPE html>
<html>
<head>
    <title>form</title>
</head>
<body>
  <form action="something_url">
    <p>
      <b>Сколько пальчев у человека?</b>
    </p>

    <p>
      <input type="radio" name="answer" value="v1">Колечно же 5<Br>
      <input type="radio" name="answer" value="v2">Всего 4<Br>
      <input type="radio" name="answer" value="v3">Нет - 3</p>
    <p>
    <input type="text" name="answer" placeholder="мой вариант">
    <input type="submit"></p>
  </form>

</body>
</html>
```

Рисунок 22 - Работа с формами

В результате должно получится:

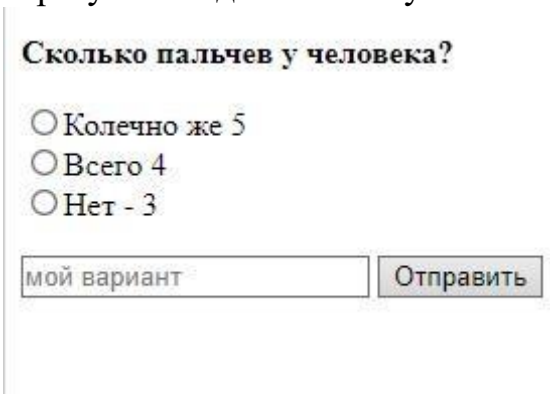


Рисунок 23 - Результат

Задание:

Изучите тег, который также относится к форме “SELECT”

Напишите форму, которая спрашивает у пользователя запрашивает персональные данные (такие как Имя, Фамилия, телефон, возраст (с помощью <select>), адрес и секретный вопрос (например: «Любимы цвет»)). Форма должна содержаться в блоке шириной 400px автоматической высотой, цветом на усмотрение, с отступом сверху в 75px и располагаться по центру экрана.

ПРАКТИЧЕСКАЯ РАБОТА №6

Создание выпадающего меню с помощью CSS

Выпадающее меню служит в качестве обзора иерархии разделов, которые содержатся в пункте меню, объединяющем их. Обычно в меню перечисляются все подразделы определенной секции, если навести указатель мыши на нее.

Выпадающее меню очень удобно, когда показывает все содержание всех секции, содержащихся на сайте, и дает возможность перейти на любую страницу из любого места сайта.

Ход работы:

1. Создайте HTML (index.htm) документ, в котором составьте разметку нашей страницы, то есть – меню с подключением

стилей

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <link rel="stylesheet" type="text/css" href="styles.css">
    </head>
    <body>
       <ul id="nav">
            <li>
                <a href="#" title="Вернуться на главную страницу">Главная</a>
            </li>
            <li>
                <a href="#" title="Информация о компании">О нас</a>
                <ul>
                    <li><a href="#">Продукты</a></li>
                    <li><a href="#">Команда</a></li>
                </ul>
            </li>
            <li>
                <a href="#" title="Что мы можем для вас сделать">Услуги</a>
                <ul>
                    <li><a href="#">Первая услуга</a></li>
                    <li><a href="#">Вторая услуга</a></li>
                    <li><a href="#">Третья услуга</a></li>
                    <li><a href="#">Четвёртая услуга</a></li>
                </ul>
            </li>
            <li>
                <a href="#" title="Наша продуктовая линейка">Продукты</a>
                <ul>
                    <li><a href="#">Первый продукт</a></li>
                    <li><a href="#">Второй продукт</a></li>
                    <li><a href="#">Третий продукт </a></li>
                    <li><a href="#">Четвёртый продукт</a></li>
                    <li><a href="#">Пятый продукт</a></li>
                    <li><a href="#">Шестой продукт</a></li>
                    <li><a href="#">Седьмой продукт</a></li>
                    <li><a href="#">Восьмой продукт</a></li>
                </ul>
            </li>
            <li>
                <a href="#" title="Как с нами связаться">Контакт</a>
                <ul>
                    <li><a href="#">Часы работы</a></li>
                    <li><a href="#">Адрес</a></li>
                </ul>
            </li>
        </ul>
    </body>
    </html>
```

Элемент #nav <ul> содержит серию элементов <li>. Все пункты, которые нуждаются в выпадающих подпунктах, содержат другой элемент <ul>. Обратите внимание, что элемент <ul> выпадающих подпунктов не имеет класса.

2. Будем использовать CSS для трансформирования серии вложенных списков <ul> в выпадающее меню. Создайте файл styles.css в корневой папке index.htm, и добавьте в него данный код:

```css
#nav{
    float:left;
    width:100%;
    list-style:none;
    font-weight:bold;
    margin-bottom:10px;
}
#nav li{
    float:left;
    margin-right:10px;
    position:relative;
    display:block;
}
#nav li a{
    display:block;
    padding:5px;
    color:#fff;
    background:#333;
    text-decoration:none;
    transition: .4s background; /* Делаем переход для плавности hover'а */
    text-shadow:1px 1px 1px rgba(0,0,0,0.75); /* Тень текста, чтобы приподнять его на немного */
    -moz-border-radius:2px;
    -webkit-border-radius:2px;
    border-radius:2px;
}
#nav li a:hover{
    color:#fff;
    background:#ffcc00;
    background:rgba(255,204,0,0.75); /* Выглядит полупрозрачным */
    text-decoration:none;  /* Убираем "декорацию" текста для избежание подчёркивания */
    transition: .4s background; /* Делаем переход для плавности hover'а */
}

/*--- ВЫПАДАЮЩИЕ ПУНКТЫ ---*/
#nav ul{
    list-style:none;
    position:absolute;
    left:-9999px; /* Скрываем за экраном, когда не нужно */
    opacity:0; /* Устанавливаем начальное состояние прозрачности */
    -webkit-transition:0.25s linear opacity; /* В Webkit выпадающие пункты будут проявляться */
}
#nav ul li{
    padding-top:1px; /* Вводим отступ между li чтобы создать иллюзию разделенных пунктов меню */
    float:none;
}
#nav ul a{
    white-space:nowrap; /* Останавливаем перенос текста и создаем многострочный выпадающий пункт */
    display:block;
```

```css
#nav li:hover ul{ /* Выводим выпадающий пункт при наведении курсора */
    left:0; /* Приносим его обратно на экран, когда нужно */
    opacity:1; /* Делаем непрозрачным */
}
#nav li:hover a{ /* Устанавливаем стили для верхнего уровня, когда выводится выпадающий список */
    background:#ffcc00;
    background:rgba(255,204,0,0.5); text-decoration:underline;
}
```

```
#nav li:hover ul a{ /* Изменяем некоторые стили верхнего уровня при выводе выпадающего пункта */
    text-decoration:none;
    -webkit-transition:-webkit-transform 0.075s linear;
}
#nav li:hover ul li a:hover{ /* Устанавливаем стили для выпадающих пунктов, когда курсор наводится на конкретный пункт */
    background:#333;
    background:rgba(51,51,51,0.5); /* Будет полупрозрачным */
    text-decoration:none;
    -moz-transform:scale(1.05);
    -webkit-transform:scale(1.05);
}
```

В первом разделе кода устанавливаем обычное горизонтальное меню.

!!! Обратите внимание, что селекторы #nav lu и #nav li a выделяют все элементы списка и ссылки в выпадающих пунктах тоже.

Следует отметить использование position:relative; для элементов списка. Таким образом, используется position:absolute; для вложенных элементов <ul>.

```
#nav ul{
    list-style:none;
    position:absolute;
    left:-9999px; /* Скрываем за экраном, когда не нужно */
    opacity:0; /* Устанавливаем начальное состояние прозрачности */
    -webkit-transition:0.25s linear opacity; /* В Webkit выпадающие пункты будут проявляться */
}
```

В данном коде устанавливаются стили для вложенных <ul> в пункт верхнего уровня. Очевидно, что нужно удалить метки пунктов списка с помощью list-style:none; и установить position:absolute; для позиционирования выпадающих подпунктов под пунктом списка, который их содержит.

Следующая строка гораздо более интересна. Обычно используется свойство display:none; для того, чтобы скрыть выпадающий пункт, когда он не используется. Но так как большинство программ для чтения с экрана игнорируют все, что имеет свойство display:none;, то использование такого метода очень нежелательно. Вместо этого используем абсолютное позиционирование <ul> для помещения его в позицию -9999px за пределами экрана, когда он не используется.

Затем следует свойство opacity:0;, для скрытия <ul>, и декларация для браузеров Webkit, для плавного вывода элемента <ul> при наведении курсора мыши.

```
#nav ul li{
    padding-top:1px; /* Вводим отступ между li чтобы создать иллюзию разделенных пунктов меню */
    float:none;
}
#nav ul a{
    white-space:nowrap; /* Останавливаем перенос текста и создаем многострочный выпадающий пункт */
    display:block;
```

```
#nav li:hover ul{ /* Выводим выпадающий пункт при наведении курсора */
    left:0; /* Приносим его обратно на экран, когда нужно */
    opacity:1; /* Делаем непрозрачным */
}
```

Здесь устанавливаются стили по умолчанию для пунктов списка и ссылок. Обратите внимание на свойство padding-top:1px; для элемента <li>. Так как все цвета устанавливаются для элементов <a>, то установка отступа в 1px для элемента <li> сдвигает элемент <a>, и, следовательно, цветную область от границы пункта списка. Таким образом, создается иллюзия, что пункты списка разделены.

Для элемента #nav ul a устанавливается свойство white-space:nowrap; для предотвращения переноса строк на другую строку.

Последняя часть кода выводит выпадающие подпункты, когда курсор мыши оказывается над соответствующим пунктом меню.

```
#nav li:hover a{ /* Устанавливаем стили для верхнего уровня, когда выводится выпадающий список */
    background:#ffcc00;
    background:rgba(255,204,0,0.5); text-decoration:underline;
}
```

#nav li:hover a определяет, что произойдет со ссылкой верхнего уровня, когда наследник будет иметь состояние *hover*:

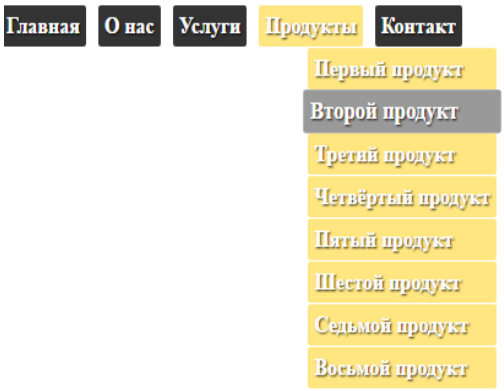
- Выпадающий список <ul> расположен внутри элемента <li>.
- Если навести курсор мыши на ссылку (<a>) в выпадающем списке (<ul>), то одновременно пункт списка верхнего уровня (<li>) тоже будет иметь состояние *hover*, так как выделен контент внутри него.
- Так как технически используется состояние *hover* для элемента списка верхнего уровня, то #nav li:hover a действует, задавая стиля для ссылки.

```
#nav li:hover ul a{ /* Изменяем некоторые стили верхнего уровня при выводе выпадающего пункта */
    text-decoration:none;
    -webkit-transition:-webkit-transform 0.075s linear;
}
```

Здесь изменяются некоторые аспекты для состояния *hover*, чтобы выпадающие элементы отличались от ссылок верхнего уровня. В данном уровне просто отключается подчеркивание текста.

Также добавляется правило для браузеров Webkit -webkit-transition:-webkit-transform 0.075s linear;, которое анимирует -webkit-transform с помощью затухания/появления в течение 0.075 секунды.

```
#nav li:hover ul li a:hover{ /* Устанавливаем стили для выпадающих пунктов, когда курсор наводится на конкретный пункт */
    background:#333;
    background:rgba(51,51,51,0.5); /* Будет полупрозрачным */
    text-decoration:none;
    -moz-transform:scale(1.05);
    -webkit-transform:scale(1.05);
}
```



В последней части кода определяются стили для выделения определенного пункта в выпадающем списке при наведении курсора мыши.

Вначале определяются два свойства background:;.

Определение background:rgba(51,51,51,0.75); устанавливает умеренно серый фон для пункта с прозрачностью 0.75. Те браузеры, которые не распознают такое определение цвета будут использовать определение цвета в старом стиле в предыдущей строке.

CSS: увеличение изображения при наведении курсора

В данной лабораторной работе рассмотрим, как создать эффект увеличения изображения при наведении на него курсором с помощью каскадных таблиц стилей.

1. Подготовьте 4 картинки (*.jpg) одинакового размера. Затем создайте HTML (index.htm) документ, в котором составьте разметку страницы.

```html
<!DOCTYPE html>
    <html>
    <head>
        <title>Увеличение картинки при наведении</title>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
        <link rel="stylesheet" type="text/css" href="styles.css">
    </head>
    <body>
        <div class="blocks"><a href="#"><img src="1.jpg" alt="thumb" class="resize_thumb"><span>
            <img src="1.jpg" alt="large">
            Здесь можно написать текст.</span></a>
        </div>
        <div class="blocks"><a href="#"><img src="2.jpg" alt="thumb" class="resize_thumb"><span>
            <img src="2.jpg" alt="large">
            Здесь можно написать текст.</span></a>
        </div>
        <div class="blocks"><a href="#"><img src="3.jpg" alt="thumb" class="resize_thumb"><span>
            <img src="3.jpg" alt="large">
            Здесь можно написать текст.</span></a>
        </div>
        <div class="blocks"><a href="#"><img src="4.jpg" alt="thumb" class="resize_thumb"><span>
            <img src="4.jpg" alt="large">
            Здесь можно написать текст.</span></a>
        </div>
    </body>
</html>
```

В данной лабораторной работе картинки вставим непосредственно в HTML код. Добавим на всякий случай ссылки к каждой картинке, чтобы картинка была интерактивной. В будущем, для <a> ссылок мы будем применять стили в CSS.

2. Далее создаем файл styles.css в корневой папке index.htm, и добавляем в него данный код:

```css
.blocks {
    float: relative;
    clear: none; /* Можно установить left или right по необходимости */
    padding-bottom: 5px; /* Расстояние между миниатюрами */
    padding-right: 5px; /* Расстояние между миниатюрами и окружающим текстом */
    margin-left:700px; /* Отступ слева для центрирования */
}
.resize_thumb {
    /* Вводим нужный размер миниатюры здесь */
    width: 150px;
    height:100px;
}
```

3. Этот стиль будет использоваться для блока миниатюры.

Теперь добавим еще стиль, он будет для <a> ссылок. Уберем все “декорации” для ссылки и курсора.

```css
.blocks a {
    display:block;
    text-decoration: none; /* Убираем подчеркивание ссылки */
    cursor:default; /* Меняем указатель мыши на обычный */
}
.blocks a:hover{
    position:relative;
}
```

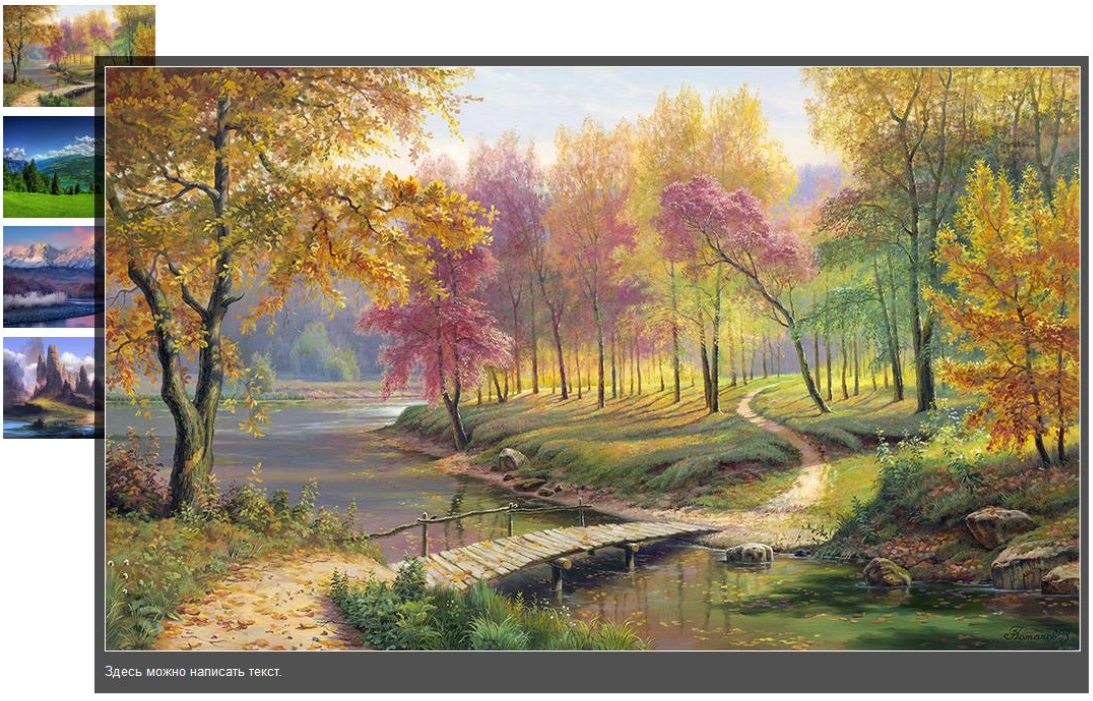
4. Следующим шагом добавим стиль для появляющейся увеличенной картинки. Описание всех строк имеется в коде.

```css
.blocks span img {
    border: 1px solid #FFFFFF; /* Добавляем рамку вокруг изображения */
    margin-bottom: 8px; /* Сдвигаем текст вниз от изображения */
    width:100%; /* Размер картинки по всей площади блока */
}

.blocks a span {
    position: absolute;
    display:none;
    color: #FFFFFF; /* Цвет названия */
    text-decoration: none; /* Убираем подчеркивание ссылки */
    font-family: Arial, Helvetica, sans-serif; /* Устанавливаем шрифты */
    font-size: 13px; /* Размер шрифта названия */
    background-color: rgba(0,0,0,0.7); /* Делаем фон появляющегося блока прозрачным */
    padding-top: 10px;  /* Вводим отступы */
    padding-right: 10px;
    padding-bottom: 13px;
    padding-left: 10px;
}
```

5. И заключающий стиль будет hover. Для того, чтобы наша увеличенная картинка появилась, нужно добавить такой стиль:

```css
.blocks a:hover span {
    display:block;
    top: 50px; /* Большое изображение появляется вверху на 50px от миниатюры */
    left: 90px; /* Большое изображение появляется справа на 90px от миниатюры */
    width:80%; /* Процентное соотношение размера блока со страницей */
    opacity:0.97; /* Небольшая прозрачность большого изображения */
    /* Если добавить правило cursor:default;, то отключится курсор в виде руки на большом изображении */
}
```

Итог:



Практическая работа №7

Основные положения JavaScript

Сценарий JavaScript может быть помещен в любом месте web страницы внутри контейнера script. Если во время интерпретации HTMLдокумента браузер встретит тег script, он первым делом выполнит код

скрипта и лишь затем продолжит интерпретацию страницы дальше.

Контейнеров script в одном документе может быть сколько угодно.

В общем виде встраивание скрипта в страницу с помощью тега script имеет вид:

```
<script type="text/javascript">
код сценария
</script>
```

Параметр type можно и не указывать, так как значение text/javascript является значением по умолчанию.
Пример web-страницы со встроенным сценарием, который выводит на экран окошко предупреждения с приветствием «Hello, World!»:

```
<html>
<head>
<title>Приветствие</title>
</head>
<body>

<script type="text/javascript">
alert('Hello, World!');
</script>
<p>Страница с приветствием</p>
</body>
</html>
```

Документ может содержать несколько тегов <script>. Все они последовательно обрабатываются интерпретатором JavaScript. В следующем примере в раздел <body> (в тело) HTML-документа вставлены операторы языка JavaScript.

Переменная – это «именованное хранилище» для данных. Мы можем использовать переменные для хранения товаров, посетителей и других данных.

Для создания переменной в JavaScript используйте ключевое слово let.

Приведённая ниже инструкция создаёт (другими словами, объявляет) переменную с именем «message»:

```
let message;
```

Теперь можно поместить в неё данные (другими словами, определить переменную), используя оператор присваивания =:

```
let message;
```

```
message = 'Hello'; // сохранить строку 'Hello' в переменной
```
с именем message

Строка сохраняется в области памяти, связанной с переменной. Мы можем получить к ней доступ, используя имя переменной:

```
let message;
message = 'Hello!';

alert(message); // показывает содержимое переменной
```

Для краткости можно совместить объявление переменной и запись данных в одну строку:

```
let message = 'Hello!'; // определяем переменную и присваиваем ей значение

alert(message); // Hello!
```

Мы также можем объявить несколько переменных в одной строке:

```
let user = 'John', age = 25, message = 'Hello';
```

Такой способ может показаться короче, но мы не рекомендуем его. Для лучшей читаемости объявляйте каждую переменную на новой строке.

Многострочный вариант немного длиннее, но легче для чтения:

```
let user = 'John';
let age = 25;
let message = 'Hello';
```

Некоторые люди также определяют несколько переменных в таком вот многострочном стиле:

```
let user = 'John',
  age = 25,
  message = 'Hello';
```

В старых скриптах вы также можете найти другое ключевое слово: var вместо let:

```
var message = 'Hello';
```

Ключевое слово var – почти то же самое, что и let. Оно объявляет переменную, но немного по-другому, «устаревшим» способом.

Пример 1. Вычисление площади треугольника

Необходимо написать сценарий, определяющий площадь прямоугольного треугольника по заданным катетам. Сценарий разместим в разделе <body> HTML-документа (листинг 1).

Листинг 1. Первый сценарий в документе :

```
<HTML>
<HEAD>
<title>Первый сценарий в документе</title>
</HEAD>
<BODY>
<P>Страница, содержащая сценарий.</P>
<script>
<!--
let a=8; h=10 /*Инициализируются две переменные*/
document.write ("Площадь прямоугольного треугольника
равна ", a*h/2,".")
   /*Для формирования вывода используется метод write
   объекта document*/
//-->
</script>
<P>Конец формирования страницы, содержащей сценарий</P>
</BODY>
</HTML>
```

// document.write javascript - выводит данные в месте расположения.

Задания

1. Проверить пример из лабораторной работы.
2. Составить сценарий, в котором вычисляется площадь круга по заданному радиусу.
3. Составить сценарий, вычисляющий гипотенузу по заданным катетам.

Функция и обработка события

Основным элементом языка JavaScript является функция. Описание функции имеет вид

```
function F (V) {S},
```

где F - идентификатор функции, задающий имя, по которому можно обращаться к функции; V - список параметров функции, разделяемых запятыми; S - тело функции, в нем задаются действия, которые нужно выполнить, чтобы получить результат. Необязательный оператор return определяет возвращаемое функцией значение. Обычно все определения и функции задаются в разделе <head> документа. Это обеспечивает интерпретацию и сохранение в памяти всех функций при загрузке документа в браузер.

В предыдущих примерах пользователю не предоставлялась возможность вводить значения, и в зависимости от них получать результат. Интерактивные документы можно создавать, используя формы. Предположим, что мы хотим создать форму, в которой поля Основание и Высота служат для ввода соответствующих значений. Кроме того, в форме создадим кнопку Вычислить. При щелчке мышью по этой кнопке мы хотим получить значение площади треугольника. Действие пользователя (например, щелчок кнопкой мыши) вызывает событие. События в основном связаны с действиями, производимыми пользователем с элементами форм HTML. Обычно перехват и обработка события задается в параметрах элементов форм. Имя параметра обработки события начинается с приставки on, за которой следует имя самого события. Например, параметр обработки события click будет выглядеть как onclick.

Листинг 1. Реакция на событие Click.

```
<HTML>
<HEAD>
<title>Обработка значений из формы</title>
<script language="JavaScript">
<!--//
function care (a, h)
{
var s=(a*h)/2;
document.write ("Площадь прямоугольного
треугольника равна ",s); return s
}
//-->
```

```
</script>
</HEAD>
<BODY>
<P>Пример сценария со значениями из формы</P>
<FORM name="form1">
Основание: <input type="text"
size=5 name="st1"><hr> Высота: <in-
put type="text" size=5
name="st2"><hr>
<input type="button" value=Вычислить
onClick="care(document.form1.st1.value,          docu-
ment.form1.st2.value)"> /*По клику мыши на кнопке в
функцию care передаются два параметра - содержимое
полей ввода*/
</FORM>
</BODY>
</HTML>
```

При интерпретации HTML-страницы браузером создаются объекты JavaScript. Взаимосвязь объектов между собой представляет иерархическую структуру. На самом верхнем уровне иерархии находится объект windows, представляющий окно браузера. Объект windows является предком или родителем" всех остальных объектов. Каждая страница кроме объекта windows имеет объект document. Свойства объекта document определяются содержимым самого документа: цвет фона, цвет шрифта и т. д. Для получения значения основания треугольника, введенного в первом поле формы, должна быть выполнена конструкция

```
document.forml.stl.value
```

т.е., говоря русским языком (при этом читаем с конца), используем данные value из поля ввода с именем st1 находящегося на форме form1 объекта document.

Пример 2. Вычисление площади квадрата.

Напишем сценарий, определяющий площадь квадрата по заданной стороне. Площадь должна вычисляться в тот момент, когда изменилось значение его стороны. Пусть форма содержит два текстовых поля: одно для длины стороны квадрата, другое для вычисленной площади. Кнопка Обновить очищает поля формы. Площадь квадрата вычисляется при возникновении события change, которое происходит в тот момент, когда значение элемента формы с именем num1 изменилось, и элемент потерял фокус. HTML-код представлен в примере 2.

Листинг 2. Реакция на событие Change

```
<HTML>
<HEAD>
<title>Обработка события Change - изменение значения
элемента</title>
<script>
function srec(obj)
{obj.res.value=obj.num1.value* obj.num1.value}
</script>
</HEAD>
<BODY>
<P>Вычисление площади квадрата</P>
<FORM name="form1">
Сторона: <input type="text" size=7 name="num1" on-
Change="srec(form1)">
<hr>
Площадь: <input type="text" size=7 name="res"><hr>
<input type="reset" value=Обновить>
</FORM>
</BODY>
</HTML>
```

Событие Focus возникает в момент, когда пользователь переходит к элементу формы с помощью клавиши <Tab> или щелчка мыши. Событие "потеря фокуса" (Blur) происходит в тот момент, когда элемент формы теряет фокус. Событие select вызывается выбором части или всего текста в текстовом поле. Например, щелкнув дважды мышью по полю, мы выделим поле, наступит событие select, обработка которого приведет к вычислению требуемого значения.

В языке JavaScript определены некоторые стандартные объекты и функции, пользоваться которыми можно без предварительного описания. Одним из стандартных объектов является объект Math. В свойствах упомянутого объекта хранятся основные математические константы, а его методы можно использовать для вызова основных математических функций.

Выражение y=log x запишется y=Math.log(x).

Форма HTML

Давайте создадим форму HTML (оставим метод action пока пустым, так как проверка данных на стороне сервера в этой лабораторной работе не раскрывается):

```
<form class="contact_form" action="" method="post" name="contact_form">
</form>
```

Элементы формы HTML

Для получения организованного и структурированного контента своей формы, обернем ее элементы (label, input и т.д.) в список. Создадим заголовок формы и первого элемента input:

```
<ul>
<li>
    <h2>Contact Us</h2>
    <span class="required_notification">* Denotes Required Field</span>
</li>
<li>
    <label for="name">Name:</label>
    <input type="text" name="name" />
</li>
</ul>
```

Подсказки для полей формы

Сделаем форматированные подсказки для полей ввода электронного адреса “email” и вебсайта “website”. Поэтому добавим свои подсказки под поля ввода, где это нужно, и назначим им класс, чтобы можно было позже определить им стили.

```
<li>
<label for="email">Email:</label>
<input type="text" name="email" />
<span class="form_hint">Proper format "name@something.com"</span>
</li>
```

Остальные элементы input

Создадим остальные элементы формы, не забывая о том, что нужно обернуть каждый раздел в элемент списка.

```
<li>
<label for="website">Website:</label>
<input type="text" name="website" />
<span class="form_hint">Proper format "http://someaddress.com"</span>
</li>
<li>
<label for="message">Message:</label>
<textarea name="message" cols="40" rows="6" ></textarea>

</li>
<li>
<button class="submit" type="submit">Submit Form</button>
</li>
```

Добавляем атрибут placeholder

Одно из первых усовершенствований, которые предлагает HTML5 для веб-форм (с которым вы, возможно, уже знакомы) – это способность установить текст-подсказку. Он показывается, когда поле ввода либо пустое, либо находится не в фокусе.

Добавим атрибут placeholder для всех текстовых тегов input. Это поможет пользователю понять, что нужно ввести в каждое поле.

```
<input type="text" name="name" placeholder="John Doe" />
<input type="text" name="email" placeholder="john_doe@example.com" />
<input type="text" name="website" placeholder="http://johndoe.com/" required/>
```

Подсказка: Назначьте placeholder'у стили

Вот вам подсказка: если нужно определить стили тексту-подсказке, к вашим услугам имеются префиксы браузеров:

```
-moz-placeholder {
    color: blue;
}
 ::-webkit-input-placeholder {
    color: blue;
}
```

В современных браузерах поддержка атрибута placeholder налажена довольно хорошо (кроме IE9, к сожалению). Если вам реально требуется поддерживать его во всех браузерах, можно посмотреть решение проблемы в javascript.

ИТОГОВАЯ РАБОТА

Проектирование структуры и информационного содержания сайта

Порядок выполнения:

Выделить основные сайты и порталы, посвященные заданной теме.

Подобрать информацию по теме сайта.

Подобрать иллюстрации и инфографику. Для каждой иллюстрации подобрать название.

Структурировать собранную информацию: составить список разделов сайта, выделить подразделы.

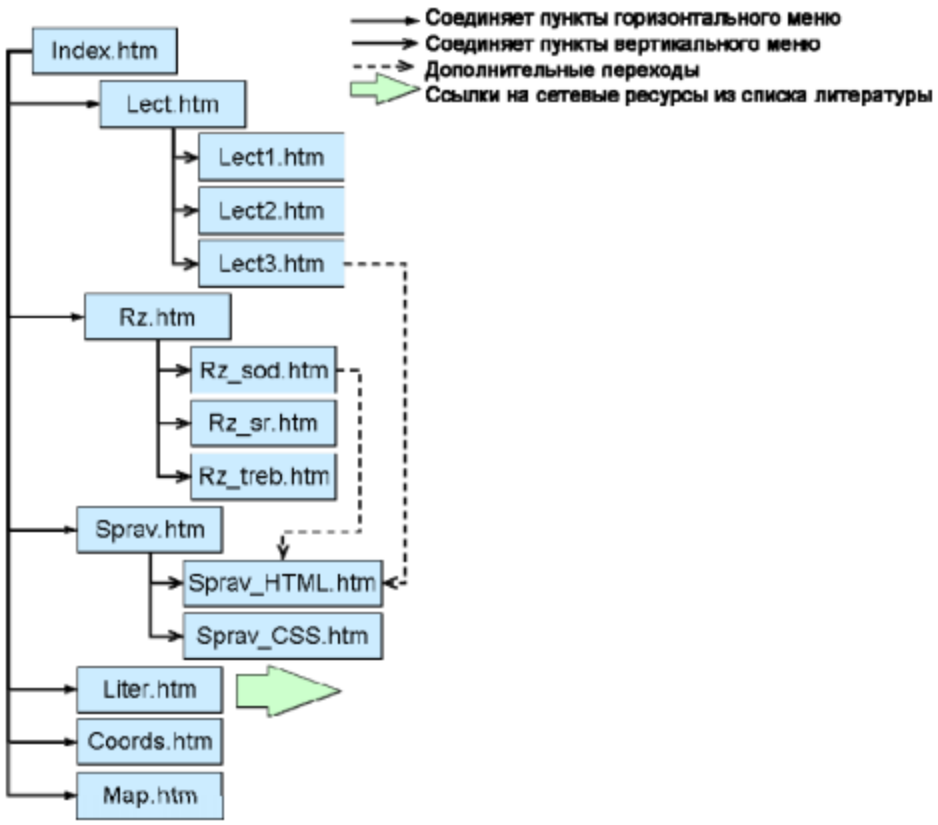
Разбить информацию на отдельные статьи.

Методические указания по выполнению

Выбор темы сайта

Пример.

Карта сайта, посвященного дисциплине «Киберпространство»:



Выбрать элементы навигации

Указания. Навигация – это перемещение по сайту. Система навигации сайта - одна из важнейших составляющих понятия " дизайн сайта". Список элементов навигации см. в лекционном материале.

Пример.

Элементы навигации для сайта, посвященного дисциплине «Киберпространство»

Основные элементы навигации:

• Горизонтальное меню для навигации по составляющим учебно- методического комплекса (лекции, расчётное задание, литература и проч.)

• Вертикальное меню для навигации по разделам лекционного материала и методическим указаниям .

Дополнительные элементы навигации:

• «Хлебные крошки»

• Переход на главную страницу под баннером

• Карта сайта

Разработать макеты страниц сайта

Указания. Дизайн-макет (или просто макет) – это рисунок, представляющий предполагаемый будущий внешний вид страниц сайта. Макеты бывают с фиксированной шириной, или с плавающей шириной. Тип макета следует указывать в отчёте. Подробнее см. в лекционном материале.

Пример.

Макет с плавающей шириной.