Лабораторная работа №1.
Основы проектирования web-сайтов.
Изучить теоретический материал

Регистратор доменных имен REG.RU: http://reg.ru

Адреса сайтов популярных хостинг-компаний:

SpaceWeb: http://sweb.ru
RU-CENTER: https://www.nic.ru
Majordomo: http://www.majordomo.ru
1Гб.ру: http://www.1gb.ru
Beget: https://beget.com
Спринтхост: http://sprinthost.ru
Мастерхост: http://masterhost.ru
TIMEWEB: http://timeweb.ru
Джино: http://jino.ru
Интернет Хостинг Центр: http://www.ihc.ru

Для того, чтобы определиться с выбором хостинга, необходимо:

1. Понять, веб-сайт с какими характеристиками вы собираетесь на нем размещать. Для тех, кто собирается разрабатывать онлайн-журнал, блог подойдет виртуальный хостинг. Первоначально выбирайте самый простой тариф, далее с ростом нагрузки вы можете перейти на другой тариф или перейти на другой хостинг. На выделенные серверы стоит переходить, когда посещаемость сайта достигнет десятков тысяч человек в сутки.
2. Оценить объем дискового пространства, которое понадобится для веб-сайта. (Например, для блога в первый год 1-2 Гбайта будет достаточно). Но необходимо учиться оптимизировать контент, чтобы страницы быстро загружались и не нагружали хостинг.
3. Узнать на каких тарифах поддерживаются программные средства, фреймворки, или CMS, которые необходимы для функционирования сайта, тип требуемой базы данных (если сайт динамический) и их версии.
4. Узнать предоставляет ли хостинг-компания тестовый период, в течение которого можно ознакомиться с функционалом, стабильностью работы хостинга, проверить запуск сайта без первоначальных вложений, оценить качество технической поддержки и удобство панели управления.
5. Узнать обеспечиваемый уровень безопасности.
6. Узнать о наличии резервного копирования.
7. Узнать расположение Data center, так как согласно закону о персональных данных все базы данных сайтов, на которых собираются персональные данные пользователей России, должны храниться на территории Российской Федерации, в противном случае доступ к веб-сайту на территории России могут заблокировать.
8. Посмотреть отзывы о хостинг-компании.

Доменное имя для сайта

Все доменные имена состоят из областей — доменов. Запись доменного имени разделяется точками. Точки разделяют уровни доменных имен, которые выделяют справа налево. По количеству этих слов определяется уровень домена. Например, доменное имя **fvte.pskgu.ru** включает в себя домен первого уровня— **ru**, далее идёт уникальный домен второго уровня **pskgu** и уже внутри своей доменной зоны **pskgu.ru** создаются сайты с доменом 3-го уровня, **fvte.**

Домены первого уровня распределяются по группам: географические домены (.ru – Россия, .рф – Россия, .ua – Украина, .by – Белоруссия, .de – Германия), тематические домены (.biz – бизнес, .museum – музеи, .info – информационные сайты, .com – любая коммерческая организация, .gov – правительственное учреждение, .edu – образовательное учреждение).

Домен 1 уровня выбирают, по нему определяют соответствующую зону, а домен 2 уровня покупают. Оплата производится за год, а далее аренду домена нужно продлевать. Домен 3 уровня и следующих уровней можно создавать самостоятельно на базе домена 2 уровня.
В интернете уже зарегистрировано огромное количество доменных имен в различных зонах, поэтому подобрать себе хорошее (короткое, простое, звучное, запоминающееся) имя в требуемой зоне в настоящее время весьма непростая задача. Первое, что стоит сделать при подборе подходящего домена для сайта — это проверить его на занятость в различных зонах. На сайте регистратора существует такая возможность.
Подобрать и зарегистрировать доменное имя, при желании, можно в любой стране. В каждой стране есть аккредитованные регистраторы доменных имен. Если Ваш проект ориентирован, например, на Европейскую аудиторию, то и регистрировать его лучше, где-то в Европе. Если проект рассчитан на Российских посетителей, то регистрировать необходимо в России. Связано это с техническими показателями. Так как проект, зарегистрированный, например, в Санкт-Петербурге, будет открываться гораздо быстрее в центральной России, чем во Владивостоке или США. А скорость загрузки сайта это один из важных факторов ранжирования.
Регистраторы представляют услуги не только напрямую, но и через хостинг-компании, поэтому адрес сайта можно подбирать через выбранного хостинг-провайдера.
Задание 1
Цель задания – получить практические навыки выбора хостинга и доменного имени.

1. Подобрать доменное имя для своего веб-сайта на сайте **Регистратора доменных имен REG.RU** в зоне RU и определить стоимость выбранного доменного имени.
2. Выбрать тариф с минимальной стоимостью и с поддержкой Python на каждом из перечисленных в Таблице 1 сайтах популярных хостинг-компаний, а также определить бесплатный тестовый период для хостинга в днях.
3. На основе данных, полученных в п.2 задания 1, заполнить Таблицу 1 (выбирая соответствующее значение из списка).

Таблица 1

№	Хостинговая компания	Название тарифа с поддержкой Python	Бесплатный тестовый период
1	REG.RU http://hosting.reg.ru		
2	SpaceWeb http://sweb.ru		
3	Beget https://beget.com		

1. Какие веб-страницы хранятся в виде файлов на дисках серверов

2. Страницы сайта доступны в Интернете средствами протоколов:

Допускается один или несколько правильных ответов

- ☐ HTTP
- ☐ HTTPS
- ☐ FTP

- [ ] SMTP

3. Служба доменных имен – это

4. Задачи веб-сервера:

Допускается один или несколько правильных ответов

- [ ] Отображение html-документов веб-сайта
- [ ] Хранение html-документов веб-сайта
- [ ] Отдача клиенту html-документов веб-сайта
- [ ] Генерация html-документов веб-сайта

5. Автономная (выделенная) часть дискового пространства на сервере и фиксированные ресурсы. Это

- ( ) Виртуальный хостинг
- ( ) Виртуальный выделенный сервер
- ( ) Выделенный сервер
- ( ) Colocation

6. Взаимодействие веб-клиента с веб-сервером происходит по протоколу:

Допускается один или несколько правильных ответов

- [ ] HTTP
- [ ] HTTPS
- [ ] FTP
- [ ] SMTP

7. Доменное имя не может содержать название брэнда

8. Веб-сайт может быть размещен в любой стране

9. Все хостинг-компании обязаны предоставлять бесплатный тестовый период

10. Серверное программное обеспечение:

Допускается один или несколько правильных ответов

- [ ] Операционная система
- [ ] Веб-сервер
- [ ] Браузер
- [ ] База данных

11. При выборе хостинга важен только поддерживаемый хостингом тип СУБД, а её версия не важна

12. Напишите прописными буквами название единообразного определителя местонахождения ресурса

13. Задачи веб-клиента:

Допускается один или несколько правильных ответов

- [ ] Загрузка html-документов
- [ ] Генерация html-документов
- [ ] Хранение html-документов
- [ ] Отображение html-документов

14. Домен какого уровня в доменном имени необходимо покупать

15. Сопоставьте название средства разработки веб-сайта с его типом:

Язык программирования

16. Система управления контентом

17. Конструктор сайтов

Лабораторная работа №2-3.
Основы создания WEB – страниц с помощью HTML

Задание 1. Вводим код. Измените в 7 строке код `<h1>Заголовок</h1>` на `<h1>Поехали!</h1>`.

```
<!DOCTYPE html>
<html>
   <head>
      <title>Поехали!</title>
   </head>
   <body>
      <h1>Заголовок</h1>
      <p>Язык HTML достаточно простой. Сначала может показаться, что в нём слишком много тегов. Но не волнуйтесь. Мы постепенно познакомимся с ними на практике. А на практике всё запоминается легко.</p>
      <p>Посмотрите на нижнюю часть мини-браузера, там вы увидите окошко с задачами, которые нужно выполнить, чтобы пройти задание.</p>
```

```
      <p>Измените строку "Заголовок" на "Поехали!" внутри h1. Если вы всё сделали правильно, то загорится кнопка "Следующее задание".</p>
   </body>
</html>
```

Задание2.
При создании веб-страниц используются два языка: HTML и CSS. HTML отвечает за структуру и содержание, а CSS — за оформление. Браузер объединяет HTML- и CSS-код и формирует внешний вид страницы.
В прошлом задании вы изменяли код в HTML-редакторе и убедились, что при этом меняется содержание страницы.
В этом задании вы поработаете с CSS-редактором и увидите, как с помощью нескольких строк кода можно изменить оформление страницы.
Сейчас мы не будем углубляться в значение каждого CSS-свойства. А просто посмотрим на CSS в действии! Чтобы включить CSS-код, нужно будет удалить символы комментариев, которые «выключают» его.

В HTML и CSS введите код. Удалите /* и посмотрите как преобразился код.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <title>CSS в действии</title>
   </head>
   <body>
      <h1>CSS</h1>
      <p>CSS ещё проще, чем HTML. Он состоит из множества правил, примерно таких:</p>
      <pre>
селектор{
   свойство1: <em>значение</em>;
   свойство2: <em>значение</em>;
}</pre>
      <p>Правила очень простые. Но есть одна проблема: свойств <em>очень</em> много.</p>
      <p>Большая часть курса будет посвящена именно CSS.</p>
      <p>А сейчас поработайте с нижним редактором.</p>
   </body>
</html>
```

Код в CSS

```
/*
body {
   padding: 0 20px;
   font-family: Arial, sans-serif;
   font-size: 16px;
}

h1 {
   color: #618AD2;
```

```
    text-shadow: 2px 2px 0 #ccc;
}

pre {
    padding: 10px;
    font-size: 14px;
    line-height: 20px;
    background: #f5f5f5;
    border: 1px solid #ccc;
    border-radius: 3px;
}

em {
    color: #618AD2;
}
```

Задание3.

Язык HTML состоит из тегов. Теги — это те самые кирпичики, из которых построена каждая веб-страница.
Каждый тег начинается с символа `<` и заканчивается символом `>`, например: `<p>`.
Все теги можно разделить на парные и одиночные. Каждый парный тег состоит из двух частей: открывающего тега и закрывающего. Внутри закрывающего тега используется символ `/`.
Вот пример парного тега:

```
<p>Абзац</p>
```

Как вы видите, у тега `<p>` есть пара в виде закрывающего тега `</p>`.

- В HTML и CSS введите код.
- В конец текста добавьте заголовок первого уровня с содержанием Потренировался.
- После него добавьте абзац с текстом Пора двигаться дальше.
- Увеличьте размер шрифта заголовков до 32px.

Код в HTML

```
<!DOCTYPE html>
<html>
    <head>
        <title>Закрепление</title>
    </head>
    <body>
        <h1>Структура HTML</h1>
        <p>Язык HTML состоит из множества тегов. Каждый тег имеет определенный смысл и предназначение.</p>
        <p>Каждый тег может иметь определенный набор атрибутов.</p>
        <h1>Структура CSS</h1>
        <p>CSS состоит из селекторов и свойств.</p>
        <p>Селекторы описывают какие именно элементы или группы элементов будут обладать заданными свойствами.</p>
    </body>
</html>
```

Код в CSS

```
/* Это правило задаёт общий размер шрифта */
body {
   font-size: 16px;
}

/* Это правило — размер для заголовков первого уровня */
h1 {
   font-size: 20px;
}

/* А это — тень для заголовков первого уровня */
h1 {
   text-shadow: 2px 2px 0 #ccc;
}
```

Задание4.

С некоторыми парными тегами мы познакомились. А что же за одиночные теги?
Парные теги обычно нужны, чтобы оформить некоторый участок текста. Благодаря паре тегов вы можете указать начало и конец этого участка. Но ведь есть теги, которые не предназначены для оформления фрагментов текста.
Например, тег для вставки изображения или тег для вставки разделительной полосы. Такие теги добавляют на страницу одиночный объект, и им не нужно для этого заключать в себя какой-то текст. Поэтому их называют одиночными.
Примеры таких тегов: `<br>`, `<hr>`, `<img>`.
Кстати, в HTML-редакторе вы увидите такие фрагменты кода: `<!-- текст -->`. Они называются «комментарии», и браузер не отображает их на странице.
Раньше одиночные теги писались с закрывающим слешом перед закрывающей скобкой. Например: `<br/>`.
В новом стандарте HTML5 использование закрывающего слеша в одиночных тегах необязательно.

- В HTML и CSS введите код.
- Оберните в `h1` заголовок `Типы тегов`.
- Оберните в отдельный тег `p` каждое предложение после заголовка.
- Оберните *одно* любое слово в тег `em`.
- Оберните *одно* любое слово в тег `strong`.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <title>Азы HTML</title>
```

```
</head>
<body>

    Типы тегов

    Каждый HTML-тег имеет свой смысл и назначение.

    Часть тегов служебные и их никогда не видно на странице.

    Другие теги используются для форматирования текста и их влияние весьма заметно.

    Например, если обернуть текст в тег h1, то он станет крупнее и вокруг него появятся отступы, так как h1 — это заголовок.



</body>
</html>
```

Код в CSS

```
/* Так можно указывать шрифт для всего документа */
body {
    font-family: Georgia, serif;
}
```

Задание5.
Парные теги обычно нужны, чтобы оформить некоторый участок текста. Благодаря паре тегов вы можете указать начало и конец этого участка. Но ведь есть теги, которые не предназначены для оформления фрагментов текста.
Например, тег для вставки изображения или тег для вставки разделительной полосы. Такие теги добавляют на страницу одиночный объект, и им не нужно для этого заключать в себя какой-то текст. Поэтому их называют одиночными.
Примеры таких тегов: `<br>`, `<hr>`, `<img>`.
Кстати, в HTML-редакторе вы увидите такие фрагменты кода: `<!-- текст -->`. Они называются «комментарии», и браузер не отображает их на странице.
Раньше одиночные теги писались с закрывающим слешом перед закрывающей скобкой. Например: `<br/>`.
В новом стандарте HTML5 использование закрывающего слеша в одиночных тегах необязательно.

- В HTML и CSS введите код.
- Вместо текста `<!-- Изображение -->` вставьте тег `<img>` (сама картинка появится в следующем задании).
- Вместо `<!-- Разделитель -->` вставьте тег `<hr>`.
- Вместо надписей `<!-- Перенос -->` вставьте теги `<br>`.
-

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <title>Одиночные HTML-теги</title>
   </head>
   <body>
      <h1>Инструктор Кекс</h1>
      <p>В последующих курсах вам будет часто помогать с освоением тонкостей HTML и CSS инструктор Кексик. Дадим же ему возможность представиться:</p>
<!-- Изображение -->
<!-- Разделитель -->
      <blockquote>
         <p>Привет! Меня зовут Кекс и я ваш будущий инструктор. Я веб-разработчик и живу в Санкт-Петербурге. Мои самые известные проекты <!-- Перенос -->
         блог Cat Energy, <!-- Перенос -->
         курс про ссылки и изображения в HTML Academy, <!-- Перенос -->
         курс про HTML5 там же.</p>
         <p>До встречи в последующих курсах!</p>
      </blockquote>
   </body>
</html>
```

Код в CSS

```
body {
   font-family: Georgia, serif;
}

/* Пример оформления цитаты */
blockquote {
   margin: 1.5em 0;
   padding: 0.5em 15px;
   line-height: 1.5;
   background: #f9f9f9;
   border-left: 2px solid #ccc;
}
```

Задание6.

- В HTML и CSS введите код.

После вставки в код тега `<img>` ничего не произошло. Почему же так вышло?
Теги могут иметь атрибуты. Некоторые теги есть смысл использовать только с атрибутами. Наиболее яркий пример — тег `<img>`, обозначающий изображение. Для него обязательно нужно указывать атрибут `src`, который задаёт адрес картинки (иначе браузер не сможет загрузить её).
В общем случае тег записывается следующим образом:

```
<имя-тега атрибут1="значение1" атрибут2="значение2" ...>
```

Атрибутов может быть несколько, вот примеры:

```
<p class="important">...</p>
```

```
<a class="external" href="https://htmlacademy.ru">...</a>
<img class="avatar" src="keks.png">
```
Не забудьте пробелы между названием тега и атрибутом и между атрибутами

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <title>Атрибуты HTML-тегов</title>
   </head>
   <body>
      <h1>Инструктор Кекс</h1>
      <p>В последующих курсах вам будет часто помогать с освоением тонкостей HTML и CSS инструктор Кексик. Дадим же ему возможность представиться:</p>
      <img>
      <hr>
      <blockquote>
         <p>Привет! Меня зовут Кекс и я ваш будущий инструктор. Я веб-разработчик и живу в Санкт-Петербурге. Мои самые известные проекты:<br>
         блог Cat Energy,<br>
         курс про ссылки и изображения в HTML Academy,<br>
         курс про HTML5 там же.</p>
         <p>До встречи в последующих курсах!</p>
      </blockquote>
   </body>
</html>
```

Код в CSS

```
body {
   font-family: Georgia, serif;
}

blockquote {
   margin: 1.5em 0;
   padding: 0.5em 15px;
   line-height: 1.5;
   background: #f9f9f9;
   border-left: 2px solid #ccc;
}

/* Пример оформления фотографии */
.photocard {
   display: block;
   width: 300px;
   margin: 20px auto;
   border-radius: 10px;
   box-shadow: 0 0 5px #666;
}
```

Задание7.

- В HTML и CSS введите код.

HTML позволяет вкладывать теги друг в друга, и одна из самых частых ошибок заключается в неправильной вложенности, например:

```
<p>Текст <strong>выделен</p> полужирным</strong>
```

В этом примере тег `<p>` закрывается раньше, чем тег `<strong>`, и это ошибка.
Другой тип ошибок случается из-за досадных опечаток и невнимательности, когда забывают пробелы между атрибутами тега или неправильно пишут их названия.

1. Обратите внимание на порядок закрытия тегов.
2. Хватает ли пробелов?
3. Проверьте правильность написания атрибутов. Их там немного.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Ищем ошибки</title>
  </head>
  <body>
    <h1>Инструктор Кекс [v.2]</h1>

    <!-- В этом блоке затерялась первая ошибка -->
    <div class="error1">
      <p><em>В последующих курсах вам будет часто помогать с освоением тонкостей HTML и CSS</p> инструктор Кексик.</em>
    </div>

    <!-- Здесь спрятана вторая ошибка -->
    <div class="error2">
      <imgsrc="/assets/course1/keks-macho.jpg" class="photocard">
    </div>

    <!-- А здесь третья -->
    <div class="error3">
      <blockquote clas="nice-cite">
        <p>Привет! Меня зовут Кекс и я ваш будущий инструктор. Я веб-разработчик и живу в Санкт-Петербурге.</p>
        <p>До встречи в последующих курсах!</p>
      </blockquote>
    </div>
  </body>
</html>
```

Код в CSS

```
body {
  font-family: Georgia, serif;
```

```
}

.photocard {
   display: block;
   width: 300px;
   margin: 20px auto;
   box-shadow: 5px 5px 0 #E7471E;
}

.nice-cite {
   margin: 1.5em 0;
   padding: 0.5em 25px;
   line-height: 1.5;
   background: #f5f5f5;
   border-left: 5px solid #E7471E;
}
```

Задание7.

- В HTML введите код.
- Сделайте цвет текста первых трёх абзацев зелёным `green`,
- В последних трёх — красным `red`.

Азы CSS

CSS — это язык для управления внешним видом HTML-документа. С помощью CSS можно задавать параметры отображения любого тега: ширину и высоту, отступы, цвет и размер шрифта, фон и так далее.

CSS это аббревиатура «Cascading Style Sheets» или «Каскадные Таблицы Стилей».

Обычно CSS называют просто «стили».

Самый простой способ применить стили к тегу заключается в использовании атрибута `style`. Например:

```
<p style="color: red;">...</p>
```

В этом примере абзацу задан красный цвет шрифта. Такой способ задания стилей называют «инлайновые стили» или «встроенные стили».

Синтаксис таких стилей очень простой: `свойство: значение;`. Причём свойств может быть несколько.

Теперь давайте составим мини-конспект раздела и отметим зелёным цветом те темы, которые мы уже отработали на практике. Для стилизации используем инлайновые стили.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <title>Азы CSS</title>
   </head>
   <body>
      <h1>Конспект курса</h1>
      <p style="color: green;">Парные теги.</p>
      <p>Одиночные теги.</p>
      <p>Атрибуты тегов.</p>
```

```
    <p>Инлайновые (встроенные) стили.</p>
    <p>Внешние стили.</p>
    <p>Стилизация по классам.</p>
  </body>
</html>
```

Задание9.

- В HTML и CSS введите код.
- Удалите все атрибуты `style` у абзацев,
- Потом удалите символы `/*` в первой строке CSS-редактора.

Задавать стили каждого тега с помощью атрибута `style` очень затратно и хлопотно. А ещё это приводит к засорению HTML-кода избыточными, повторяющимися кусками CSS.
К счастью, есть и другие способы подключения стилей. Первый — подключение внешнего файла с помощью тега `<link>`, а второй — использование специального тега `<style>`. Подробнее эти методы будут разобраны в последующих разделах.
А сейчас вы познакомитесь со вторым разделом редактора, помеченным как CSS. Код из CSS-редактора подставляется в HTML-документ так, как будто вы записали его в тег `<style>`.
Для начала мы почистим код в HTML-редакторе, а потом начнём пошагово стилизовать наш мини-конспект с помощью других возможностей CSS.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Другие способы подключения CSS</title>
  </head>
  <body>
  <h1>Конспект курса</h1>
    <p style="color: green;">Парные теги.</p>
    <p style="color: green;">Одиночные теги.</p>
    <p style="color: green;">Атрибуты тегов.</p>
    <p style="color: green;">Инлайновые (встроенные) стили.</p>
    <p style="color: red;">Внешние стили.</p>
    <p style="color: red;">Стилизация по классам.</p>
  </body>
</html>
```

Код в CSS

```
/*
body {
  font-family: Tahoma, serif;
}

p {
  color: green;
}
```

Задание10.

- В HTML и CSS введите код.
- Для `h1` задайте `color: #999999;`.
- Для `strong` задайте `color: green;`.
- Для `em` задайте `color: red;`.
-

Селекторы в CSS

Когда вы задаёте стили тега с помощью атрибута `style`, браузер сразу же понимает, к какому именно тегу применить эти стили. Но когда стили подключаются с помощью внешнего файла или через тег `<style>`, браузер ищет стилизуемые теги с помощью «селекторов».

С селекторами вы уже немного знакомы: в предыдущем задании вы использовали селектор `p`, который находился перед фигурными скобочками в CSS-коде. В общем случае синтаксис CSS-правил выглядит так:

```
селектор {
  свойство1: значение1;
  свойство2: значение2;
  ...
}
```

Язык селекторов очень мощный и гибкий. Простейший тип селекторов — селекторы по имени тега: `p`, `h1` и так далее. Когда браузер видит такой селектор, он применяет стили из правила ко всем подходящим тегам. Например, ко всем абзацам или ко всем заголовкам первого уровня.

Но, Хьюстон, у нас проблемы с конспектом! Селектор `p` подсветил *все* абзацы зелёным. А они должны быть разного цвета. Как быть?

Давайте попробуем добавить внутрь абзацев разные теги для разных цветов. Тогда в CSS можно будет использовать разные селекторы.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <title>Селекторы в CSS</title>
   </head>
   <body>
      <h1>Конспект курса</h1>
      <p><strong>Парные теги.</strong></p>
      <p><strong>Одиночные теги.</strong></p>
      <p><strong>Атрибуты тегов.</strong></p>
      <p><strong>Инлайновые (встроенные) стили.</strong></p>
      <p><em>Внешние стили.</em></p>
      <p><em>Стилизация по классам.</em></p>
   </body>
</html>
```

Код в CSS

```
body {
   font-family: Tahoma, serif;
}
```

/* Небольшая помощь. Дальше сами.

```
h1 {
   color: #999999;
}
```

Задание11.

- В HTML и CSS введите код.
- Для начала удалите из HTML-кода все `strong` и `em`, чтобы внутри `p` остался только текст.
- Первым четырём абзацам добавьте класс `learned-ok`.
- Пятому абзацу добавьте класс `learning-in-progress`.
- Шестому абзацу добавьте класс `not-learned`.
-

Классы в CSS

Класс — это всего лишь один из атрибутов HTML-тегов, например:

```
<p class="important">...</p>
<p class="help">...</p>
```

В CSS можно задавать стили только для элементов с определённым классом. Для этого используется селектор по классу, который пишется так `.имя-класса`, например:

```
.important { color: red; } — выберет все теги с классом "important"
.help { color: green; }    — выберет все теги с классом "help"
```

Классы гибкие, их можно создавать много и называть понятными именами. Например, можно создать класс, который отмечает раздел курса, который сейчас изучается.

Имя класса может содержать в себе латинские символы, цифры, символ дефиса `-` и подчёркивания `_` и начинаться оно должно с латинского символа.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <title>Классы в CSS</title>
   </head>
   <body>
      <h1>Конспект курса</h1>
      <p><strong>Парные теги.</strong></p>
      <p><strong>Одиночные теги.</strong></p>
      <p><strong>Атрибуты тегов.</strong></p>
      <p><strong>Инлайновые (встроенные) стили.</strong></p>
      <p><em>Внешние стили.</em></p>
      <p><em>Стилизация по классам.</em></p>
   </body>
</html>
```

Код в CSS

```
body {
   font-family: Tahoma, serif;
}
```

```
h1 {
   color: #999999;
}

strong {
   color: green;
}

em {
   color: red;
}
.learned-ok {
   color: green;
}
.learning-in-progress {
   color: orange;
}
.not-learned {
   color: red;
}
```

Задание12.

- В HTML и CSS введите код.

1. Добавьте в правило `.learned-ok` свойство `text-decoration` со значением `line-through`,
2. **Цель 2**в правило `.learning-in-progress` свойство `padding-left` со значением `15px`,
3. **Цель 3**в правило `.not-learned` свойство `background-color` со значением `#FFF0F0`.
4. **Цель 4**И напоследок отметьте последние два пункта конспекта как пройденные, заменив их класс на `learned-ok`.

Свойства и значения CSS

Селекторы указывают на то, к каким элементам применять стили, а свойства — на то, как именно отображать элементы.

Существует огромное количество CSS-свойств, которые влияют практически на все аспекты отображения элементов. Причём каждому свойству соответствует определённый набор значений.

Некоторые значения задаются с помощью текстовых констант, например `red`, `bold`, другие с помощью цифровых значений: `12px`, `120%` и так далее.

Мощь стилей заключается в том, что вы можете быстро и гибко менять внешний вид нужных элементов, особенно когда используете классы. Например, чтобы зачеркнуть текст всех изученных пунктов конспекта, нужно добавить всего лишь одну строчку в CSS:

```
.learned-ok {
  color: green;
  text-decoration: line-through;
}
```

И все теги с классом `.learned-ok` будут отображаться с перечёркнутым текстом. Теперь представьте, как долго делать то же самое через инлайновые стили, когда в конспекте больше сотни пунктов.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <title>Свойства и значения CSS</title>
   </head>
   <body>
      <h1>Конспект курса</h1>
      <p class="learned-ok">Парные теги.</p>
      <p class="learned-ok">Одиночные теги.</p>
      <p class="learned-ok">Атрибуты тегов.</p>
      <p class="learned-ok">Инлайновые (встроенные) стили.</p>
      <p class="learning-in-progress">Внешние стили.</p>
      <p class="not-learned">Стилизация по классам.</p>
   </body>
</html>
```

Код в CSS

```
body {
   font-family: Tahoma, serif;
}

h1 {
   color: #999999;
}

.learned-ok {
   color: green;
}

.learning-in-progress {
   color: orange;
}

.not-learned {
   color: red;
}
```

Задание13.

- В HTML и CSS введите код.

Помогите Кексу исправить ошибки в стилях конспекта:

- Первую ошибку с потерянными стилями для всех абзацев,
- вторую ошибку с отсутствием зачёркивания пройденных,
- третью ошибку с пропавшим фоном непройденных.
-

Работа над ошибками

Синтаксис CSS намного проще синтаксиса HTML, но это не мешает совершать досадные ошибки.
Конечно, одни из самых распространённых ошибок — это опечатки в названиях свойств или селекторах.
К другим частым ошибкам относится отсутствие `;` в списке CSS-свойств. Когда после пары «свойство-значение» забывают поставить точку с запятой, браузер не применяет ни это свойство, ни все последующие.
Когда вёрстка конспекта была почти завершена, к компьютеру шерстяной молнией подлетел инструктор Кекс и решил напоследок поиграться со стилями, чтобы сделать конспект «симпатичнее». Получилось неплохо, но в этот раз Кекс был спросонья и наделал досадных опечаток, которые сломали некоторые стили.
Ну и замечательно! Теперь есть возможность потренироваться в поиске ошибок в CSS!

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Работа над ошибками</title>
  </head>
  <body>
    <h1>Конспект курса</h1>
    <p class="learned-ok">Парные теги.</p>
    <p class="learned-ok">Одиночные теги.</p>
    <p class="learned-ok">Атрибуты тегов.</p>
    <p class="learned-ok">Инлайновые (встроенные) стили.</p>
    <p class="learning-in-progress">Внешние стили.</p>
    <p class="not-learned">Стилизация по классам.</p>
  </body>
</html>
```

Код в CSS

```
body {
  font-family: Georgia, serif;
}

p {
  margin: 10px 0;
  padding: 5px 10px
  color: white;
  border: 1px solid #ccc;
  border-left-width: 10px;
}

.learned-ok {
```

```
    text-decoration: linr-through;
    border-color: #27ae60;
    background-color: #2ecc71;
}

.learning-in-progress {
    border-color: #f39c12;
    background-color: #f1c40f;
}

.not-learned {
    border-color: #c0392b;
    backround-color: #e74c3c;
}
```

Задание14.

- В HTML и CSS введите код.

Первое испытание

Подсказка: CSS менять не надо. Достаточно использовать правильные теги и классы.

Код в HTML

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Первое испытание</title>
    </head>
    <body>
        Введение В этом пособии изучается работа с HTML и CSS. Глава 1 Работа с HTML, знакомство с тегами и атрибутами тегов. Глава 2 Работа с CSS, знакомство с селекторами и свойствами. Самое лучшее онлайн-пособие
    </body>
</html>
```

Код в CSS

```
/* Классы для оформления */
.cite {
    padding: 5px 10px;
    background: #f5f5f5;
    border-left: 2px solid #ccc;
    font-style: italic;
}
```

```
/* Служебные стили */
body {
   width: 260px;
   margin: 0;
   padding: 0 20px;
   font-family: "Arial", sans-serif;
   font-size: 12px;
   line-height: 1.4;
   color: black;
}

h1 {
   font-weight: bold;
   font-size: 2.5em;
   margin-top: 0.5em;
   margin-bottom: 0.5em;
}

h2 {
   font-size: 1.5em;
   color: #333;
   font-weight: normal;
   margin-top: 0.5em;
   margin-bottom: 0.5em;
}

p {
   margin: 1em 0;
   color: #666;
}
```

Лабораторная работа №4-5.
Скрипты. Подключение скриптов.

Задание 1

- Измените доктайп HTML на современный.

Теоретический материал

Каждый HTML-документ должен начинаться с декларации типа документа или «доктайпа». Тип документа нужен, чтобы браузер мог определить версию HTML и правильно отобразить страницу.
Например, для старой версии HTML 4.01 доктайп выглядит так:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
```

А для последней версии HTML уже намного проще:

```
<!DOCTYPE html>
```

Последнюю версию HTML ещё называют HTML 5. Но так как эта версия уже принята как стандарт и распространена почти везде, то её называют просто HTML.

Код в HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

Код в CSS

```
body {
   min-height: 500px;
   background-color: #2f358f;
   background-image: url("https://roskav.ru/wp-content/uploads/2018/01/emblema-AGPU-prozrachnaya-964x1024.png");
   background-repeat: no-repeat;
   background-size: 200px;
   background-position: 50% 50%;
}
```

Задание 2

- Добавьте текст внутрь тега `<body>`. Как минимум 10 символов.

Теоретический материал

Простейшая HTML-страница]

Простейшая HTML-страница состоит как минимум из трёх тегов.

Тег `<html>` — это контейнер, в котором находится всё содержимое страницы, включая теги `<head>` и `<body>`. Как правило, тег `<html>` идёт в документе вторым после доктайпа.

Тег `<head>` предназначен для хранения других элементов, цель которых — помочь браузеру в работе с данными. Содержимое этого тега не отображается напрямую.

Тег `<body>` предназначен для хранения содержания веб-страницы (контента), отображаемого в окне браузера.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head></head>
   <body>
      <!-- Содержание страницы -->
   </body>
</html>
```

Код в CSS

```
body {
   font-size: 52px;
   font-family: Monaco, Courier, monospace;
}
```

Задание3

- Измените заголовок страницы с `Просто Кексик` на `Инструктор Кекс`.

Теоретический материал

Заголовок HTML-страницы

Заголовок страницы — это тот текст, который отображается в левом верхнем углу браузера, а также во вкладках.

Чтобы задать заголовок страницы, нужно использовать тег `<title>` внутри тега `<head>`. Например, вот так:

```
<head>
  <title>Группа ИВТ</title>
</head>
```

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Просто Кексик</title>
  </head>
  <body>
    <h1>Тег &lt;title&gt;</h1>
    <img src=" https://avatars.mds.yandex.net/get-pdb/1050037/a6f8fe05-325f-4e40-9d98-0213e4c152fe/s1200?webp=false" alt="">
    <p>Этот тег обозначает заголовок страницы.</p>
    <p>Он позволяет очень удобно ориентироваться между множеством открытых вкладок.</p>
    <p>Инструктор Кекс рекомендует использовать понятные заголовки.</p>
  </body>
</html>
```

Код в CSS

```
h1 {
  font-size: 28px;
  font-family: Monaco, Courier, monospace;
  color: #618ad2;
}

img {
  float: right;
  width: 150px;
  margin: 0 10px;
  border-radius: 50px;
}
```

Задание 4

- В коде кодировка задана неверно, поэтому текст в мини-браузере отображается неправильно.
- Установить правильную кодировку: `utf-8`.

Теоретический материал

Кодировка HTML-страницы

Кодировку HTML-страницы нужно указывать для того, чтобы веб-браузер мог правильно отображать текст на странице. Если браузер неправильно «угадает» кодировку, то вместо текста будут отображаться иероглифы.

Чтобы сообщить браузеру кодировку HTML-страницы, необходимо внутри тега `<head>` использовать тег:

```
<meta charset="имя кодировки">
```

Самая распространённая современная кодировка — `utf-8`. Использовать её необходимо во всех проектах.

Для кириллицы в Windows `charset` часто задавали как `windows-1251`. Но сейчас это считается плохой практикой.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="windows-1251">
      <title>Кодировка HTML-страницы</title>
   </head>
   <body>
      <h1>Кодировка</h1>
      <p>Когда кодировка документа задана неверно, некоторые символы отображаются как «иероглифы», а некоторые нет.</p>
   </body>
</html>
```

Код в CSS

```
h1 {
   font-size: 28px;
   font-family: Monaco, Courier, monospace;
   color: #618ad2;
}
```

Задание 5

- Изменить ключевые слова `классы, стили, свойства` на более подходящие: `мета-теги, поисковые системы, ключевые слова`.

Ключевые слова

Есть целое семейство тегов `<meta>`, называемых мета-тегами. Их можно использовать внутри тега `<head>`.

Мета-теги различаются набором атрибутов и их значений, вот некоторые из атрибутов: `content`, `http-equiv`, `name` и `scheme`.

Мета-теги хранят полезную для браузеров и поисковых систем информацию. Один из таких тегов — это описание ключевых слов страницы. Задаётся он так:

```
<meta name="keywords" content="разные, ключевые, слова">
```

В атрибуте `content` через запятую перечисляются самые важные слова из содержания страницы. Раньше этот тег был очень важен для поисковиков. Каково положение дел сейчас — большой секрет Яндекса и Гугла.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Ключевые слова</title>
      <meta name="keywords" content="классы, стили, свойства">
   </head>
   <body>
      <h1>&lt;meta name="keywords"&gt;</h1>
      <p>Задаёт ключевые слова.</p>
```

Этот мета-тег поисковые системы используют для того, чтобы определить релевантность ссылки. При формировании тега необходимо использовать только те слова, которые содержатся в самом документе.
</body>
</html>

Код в CSS

```
h1 {
  font-size: 28px;
  font-family: Monaco, Courier, monospace;
  color: #618ad2;
}
```

Задание 6

- Изменить мета-описание страницы с `Не пересказать` на более полезное и ёмкое: `Как поисковые системы используют мета-описание страницы`.

Теоретический материал

Описание содержания страницы

Ещё один полезный для поисковых систем мета-тег — краткое описание страницы. Оно задаётся так:

```
<meta name="description" content="краткое описание">
```

В атрибуте `content` должно быть краткое содержание или аннотация страницы. Оно часто используется поисковиками при отображении результатов поиска.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Описание содержания страницы</title>
    <meta name="keywords" content="мета-теги, поисковые системы, ключевые слова">
    <meta name="description" content="Не пересказать">
  </head>
  <body>
    <h1>&lt;meta name="description"&gt;</h1>
    <p>Этот тег используется при создании краткого описания страницы, которое используется поисковыми системами во время индексации. Потом вы можете увидеть его в поисковой выдаче.</p>
    <p>При отсутствии тега поисковые системы выдают в аннотации первую строку документа или отрывок, содержащий ключевые слова.</p>
  </body>
</html>
```

Код в CSS

```
h1 {
  font-size: 28px;
  font-family: Monaco, Courier, monospace;
  color: #618ad2;
}
```

Задание 7

1. Раскомментировать оба тега `<p>` в разделе `Комментарии для пояснений`.
2. Закомментировать любой `<p>` в разделе `Или когда удалять жалко`.

Теоретический материал

HTML-комментарии

Комментарий в HTML-коде задаётся так:

<!-- любой текст -->

Текст внутри комментария не отображается браузером на странице. Комментарии обычно используются в следующих случаях:

- Для комментирования кода. Всегда полезно оставить подсказку.
- Для временного отключения кода. Удалять код неудобно, так как его надо будет восстанавливать, а закомментировать и потом раскомментировать — самое лучшее решение.

Комментарии можно использовать в любом месте страницы, кроме тега `<title>` — внутри него они не работают.

Чтобы быстро закомментировать или раскомментировать строку кода в HTML или CSS редакторе, можете использовать сочетание клавиш **ctrl + /** или **cmd + /**.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>HTML-комментарии</title>
  </head>
  <body>
    <div id="section-1">
      <h1>Комментарии для пояснений</h1>
      <!--<p>Бывает, что в HTML-коде есть сложные участки, назначение которых не совсем понятно.</p>-->
      <!--<p>В этом случае можно использовать комментарии, чтобы пояснить другому разработчику некоторые тонкости.</p>-->
    </div>
    <div id="section-2">
      <h1>Или когда удалять жалко</h1>
      <p>Иногда какой-то код очень жалко или страшно удалять.</p>
      <p>Поэтому его могут временно закомментировать.</p>
    </div>
  </body>
</html>
```

Код в CSS

```
h1 {
  font-size: 28px;
  font-family: Monaco, Courier, monospace;
  color: #618ad2;
}
```

Задание 8

- Раскомментировать тег `<style>`.

Теоретический материал

Подключение стилей

CSS-стили можно писать внутри HTML-кода страницы или подключать их как внешний файл.

В первом случае стили называются «встроенными» или «инлайновыми», а писать их нужно внутри тега `<style>`. Этот тег обычно размещают внутри `<head>`. Например:

```
<head>
  <style>
    CSS-код
  </style>
</head>
```

Внутри `<style>` пишут обычный CSS-код.

Инлайновые стили используют не так часто, например, для оптимизации скорости загрузки страницы. Чаще используют внешние стили.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Подключение стилей</title>
      <!--<style>
         h1 {
            font-size: 28px;
            font-family: Monaco, Courier, monospace;
            color: #618ad2;
         }
      </style>-->
   </head>
   <body>
      <h1>Подключение стилей</h1>
      <p>Подключение стилей можно выполнять разными способами.</p>
      <p>Один из них — описывать стили внутри HTML-страницы. Для этого используется тег &amp;lt;style&amp;gt;.</p>
   </body>
</html>
```

Задание 9

- Раскомментировать стили в CSS-редакторе.

Теоретический материал

Тайна CSS-редактора

CSS-код из редактора незаметно добавляется внутрь тега `<style>`, а этот тег добавляется в мини-браузер.

В этом задании CSS-стили такие же, как и в предыдущем, но вынесены в CSS-редактор и закомментированы. Комментарии в CSS работают так же, как в HTML — позволяют временно отключить какой-то кусок кода.

CSS-комментарии задаются с помощью символов `/*` и `*/`:

```
/*
h1 {
```

```
  color: red;
}
*/
```

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Тайна CSS-редактора</title>
   </head>
   <body>
      <h1>Тайна CSS-редактора</h1>
      <p>Наш мини-браузер использует тот код, который вы пишете в CSS-редакторе, и незаметно добавляет его в HTML-код страницы с помощью тега &lt;style&gt;.</p>
      <p>Поэтому ему не нужны никакие внешние файлы. Хотя их он тоже может использовать.</p>
   </body>
</html>
```

Код в CSS

```
/*
h1 {
   font-size: 28px;
   font-family: Monaco, Courier, monospace;
   color: #618ad2;
}
*/
```

Задание 10

- Добавить внутрь `<head>` тег `<link>`,
- Задать ему атрибут `rel` со значением `stylesheet`
- Задать атрибут `href` со значением `style.css`.

Теоретический материал

Подключение внешних стилей

Чаще всего стили подключают из внешнего файла с расширением `.css`. Для этого используется тег `<link>`. Например:

```
<head>
<link href="style.css" rel="stylesheet">
</head>
```

В атрибуте `href` задают адрес файла, а атрибут `rel="stylesheet"` говорит браузеру, что мы подключаем стили, а не что-то другое.

Лучше подключать стили внутри `<head>`, но это необязательно. Тег `<link>` будет работать и в другом месте страницы.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
```

```
    <title>Подключение внешних стилей</title>
    <!-- подключите внешние стили -->
  </head>
  <body>
    <h1>Внешние стили</h1>
    <p>Внешние стили намного удобнее встроенных, так как вы можете подключить один и тот же файл стилей ко множеству страниц.</p>
    <p>Если понадобится внести в стили изменения, то вы меняете один файл, а изменения появляются на всех страницах, где он подключен.</p>
    <p>Со встроенными стилями в этом случае пришлось бы повозиться.</p>
  </body>
</html>
```

Задание 11

- Раскомментировать тег `<script>`.
- Посмотреть, как подключенный скрипт оживит документ.

Теоретический материал

Подключение скриптов

В вебе следующее разделение ролей: HTML отвечает за структуру документа, стили — за его внешний вид, а скрипты — за поведение. С помощью скриптов, например, можно «оживлять» страницу, добавляя анимацию и другие эффекты. Скрипты создаются с помощью языка JavaScript.

Скрипты подключаются так же, как и стили: их либо пишут внутри страницы, либо подключают как внешние файлы.

Встроенные скрипты пишут внутри тега `<script>`. Например:

```
<script>
JavaScript-код
</script>
```

Тег `<script>` можно использовать в любом месте HTML-документа, но лучше вставлять его в самом конце перед закрывающим тегом `</body>`.

Часть возможностей JavaScript постепенно переходит в CSS, например, возможность задавать плавное изменение значений свойств.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Подключение скриптов</title>
  </head>
  <body>
    <h1>Подключение скриптов</h1>
    <p id="blinking">Подключение скриптов можно выполнять разными способами. Один из способов заключается в описании скриптов прямо внутри HTML-страницы.</p>
    <!--<script>
      var p = document.getElementById("blinking");
      setInterval(function() {
        if (p.style.fontSize != "10px") {
          p.style.fontSize = "10px";
        } else {
```

```
            p.style.fontSize = "20px";
          }
        }, 2000);
      </script>-->
  </body>
</html>
```

Код в CSS

```
h1 {
  font-size: 28px;
  font-family: Monaco, Courier, monospace;
  color: #618ad2;
}

p {
  transition: font-size .5s;
}
```

Задание 12

1. Перед закрывающим тегом `</body>` вставить тег `<script>`.
2. Добавить атрибут `src` со значением `scripts.js`.

Теоретический материал

Подключение внешних скриптов

Скрипты чаще всего подключают из внешних файлов с расширением `.js`. Для этого используют тег `<script>` с атрибутом `src`, в котором указывается путь к файлу. Например:

```
<script src="scripts.js"></script>
```

Обратить внимание, что тег `<script>` парный. Если подключать внешние скрипты, то просто ничего не пишут внутри тега.

Внешние скрипты лучше подключать перед закрывающим тегом `</body>`.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Подключение внешних скриптов</title>
  </head>
  <body>
    <h1>Внешние скрипты</h1>
    <p>Внешние скрипты так же, как и внешние стили, используются намного чаще встроенных.</p>
    <p>В них удобнее вносить изменения, особенно когда один и тот же скрипт подключён к большому количеству страниц.</p>
    <p>А ещё внешние ресурсы кешируются браузером, что позволяет ускорить загрузку страниц. А в вебе это важно.</p>
```

```
    <!-- подключите внешний скрипт -->
  </body>
</html>
```

Код в CSS

```
h1 {
  font-size: 28px;
  font-family: Monaco, Courier, monospace;
  color: #618ad2;
  transition: color .5s;
}
```

Задание 13

В задании закрепляем навыки описания мета-информации:

- заголовка страницы;
- ключевых слов;
- краткого описания страницы.

- Заголовок страницы `Вёрстка`.
- Ключевые слова `вёрстка, компьютерная грамотность, программирование`.
- Краткое описание `Примеры того, что изучение веб-технологий может быть полезно для всех`.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>Почему всё больше «не-технарей» изучают вёрстку</h1>

    <p>Раньше термин «компьютерная грамотность» по своей сути означал умение работать с офисным пакетом, но жизнь не стоит на месте, и теперь всё больше экспертов заявляют о том, что крайне важным навыком для широкого круга людей становится умение программировать.</p>

    <p>При этом программирование — огромная область знаний, только небольшая часть которой может действительно пригодиться обычному человеку (то есть «не-технарю»). Мы считаем, что основу «новой компьютерной грамотности» должно составить знание вёрстки. В сегодняшнем топике мы рассмотрим примеры того, как изучение веб-технологий может помочь представителям не-технических профессий.</p>

    <p>Прочитать <a href="http://habrahabr.ru/company/htmlacademy/blog/252169/" target="_blank" rel="nofollow">статью целиком</a> вы можете на Хабре.</p>
  </body>
```

```
</html>
```

Код в CSS

```
h1 {
  font-size: 28px;
  font-family: Georgia, serif;
}

p {
  line-height: 1.5;
}
```

Задание 14

1. Подключите стилевой файл `final.css`.
2. Затем подключите скрипт `final.js`.

Во втором итоговом задании вам нужно самостоятельно подключить к странице внешние ресурсы:

- Стилевой файл /assets/course2/final.css.
- Файл со скриптами /assets/course2/final.js.

Не забывайте, что хорошим тоном считается подключать внешние стили в `<head>`, а скрипты перед закрывающим тегом `</body>`.

В подключаемых файлах — демонстрация возможностей CSS и JavaScript, которые мы будем изучать в дальнейшем.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Второе итоговое задание</title>
  </head>
  <body>
    <div class="scene">
      <div class="pot pot-bot">
        <div class="shadow"></div>
        <div class="pot pot-shadow"></div>
        <div class="pot pot-top"></div>
        <div class="plant">
          <div class="head">
            <ul>
              <li></li><li></li><li></li><li></li><li></li>
              <li></li><li></li><li></li><li></li><li></li>
            </ul>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>
```

Код в CSS

```
li {
   list-style: none;
}
```

Лабораторная работа 6-7
Создание ссылок

Задание 1.

Добавьте с помощью тега `<a>` ещё две ссылки с текстом внутри. Пока что не добавляйте к этим тегам никаких атрибутов.

Теоретический материал

Что такое ссылка?

Ссылка обычно выглядит как подчёркнутый участок текста, щёлкая на который вы переходите на другую страницу, открываете изображение или начинаете скачивать файл. Если представить, что интернет это огромная сеть из множества узлов, то ссылки будут нитками, соединяющими все узлы этой сети.

Ссылки создаются с помощью очень простого и короткого тега `<a>`. Например, вот так:

```
<a href="https://htmlacademy.ru">HTML Academy</a>
```

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Что такое ссылка?</title>
   </head>
   <body>
      <h1><a href="https://htmlacademy.ru">HTML Academy</a></h1>

      Курсы<br>

      Профиль
   </body>
</html>
```

Задание 2.

- Добавьте к первой ссылке без адреса атрибут `href` со значением `https://htmlacademy.ru/courses`,
- Добавьте ко второй ссылке атрибут `href` со значением `/profile`

Теоретический материал

Задаём адрес ссылки

Тег `<a>` без адреса бесполезен, так как он описывает ссылку, которая никуда не ведёт. Поэтому в предыдущем задании текст никак не изменялся, даже после добавления тегов.

Адрес ссылки задаётся с помощью атрибута `href`:

```
<a href="http://keksby.ru">The Great Keksby</a>
```

У ссылки в примере задан адрес `http://keksby.ru`.
Значением атрибута является URL, который обычно называют просто *адрес*. Адреса бывают разные: абсолютные, относительные, указывающие на страницу, на файл, изображение, содержащие якорь и так далее. А значит и ссылка может указывать на любой объект в интернете.
Кстати, вы уже использовали URL, когда подключали внешние файлы стилей в разделе «Структура HTML-документа».

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Задаём адрес ссылки</title>
  </head>
  <body>
    <h1><a href="https://htmlacademy.ru">HTML Academy</a></h1>

    <a>Курсы</a><br>

    <a>Профиль</a>
  </body>
</html>
```

Задание 3.

- Щёлкните в мини-браузере по ссылке с адресом `https://npoed.htmlacademy.ru/htmlcss102`,
- а затем по ссылке с адресом `/htmlcss102`.
- Убедитесь, что обе ссылки ведут на одну и ту же страницу.
- Для того, чтобы вернуть содержимое мини-браузера после перехода по ссылке, нажмите кнопку

Теоретический материал

Абсолютные адреса
Поговорим поподробнее об адресах. Они могут быть абсолютными и относительными. Абсолютные адреса содержат в себе протокол, имя сервера и путь. Например, в адресе `https://htmlacademy.ru/courses`:

https:// — это *протокол*
htmlacademy.ru — *имя сервера*, также называется *домен* или *хост*
/courses — *путь*

Абсолютный адрес хорош тем, что однозначно указывает расположение документа. Браузер просто запрашивает по указанному протоколу с указанного сервера документ с указанным путём.
Иногда абсолютные адреса записываются в укороченном виде, например вот так: `/courses`.
В этом случае, браузер подставляет протокол и сервер текущей страницы. Например, если на сайте `https://htmlacademy.ru` есть ссылка с адресом `/courses`, то браузер для запроса преобразует её в такую: `https://htmlacademy.ru/courses`.

Используйте укороченные абсолютные адреса при задании ссылок внутри своего сайта, так как в случае изменения домена сайта вам не придётся ничего менять.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Абсолютные адреса</title>
  </head>
  <body>
    <h1><a href="https://htmlacademy.ru">HTML Academy</a></h1>

    <a href="https://npoed.htmlacademy.ru/htmlcss102">«Веб-программирование»</a><br>

    <a href="/htmlcss102">Тоже «Веб-программирование»</a>
  </body>
</html>
```

Задание 4.

1. Чтобы узнать, куда ведут относительные адреса, щёлкните в мини-браузере по первой ссылке,
2. второй ссылке,
3. третьей ссылке.

Чтобы вернуть содержимое мини-браузера после перехода по ссылке, нажмите кнопку

Теоретический материал

Относительные адреса

В относительных адресах нет ни протокола, ни имени сервера, а путь не начинается со слэша `/`. Вот примеры относительных адресов:

```
courses/1
./courses
../../run/1
```

В относительных адресах могут использоваться специальные символы, аналогичные символам в путях файловых систем: `.` и `..`.

Если браузер видит, что у ссылки задан относительный адрес, то он должен преобразовать этот адрес в абсолютный, чтобы знать, куда ведёт ссылка. Для этого браузер использует текущий адрес страницы. Например, так преобразуются адреса разных ссылок на одной и той же странице:

Текущий адрес	Адрес в ссылке	Преобразуется в
http://site.ru/news/1	2	http://site.ru/news/2
http://site.ru/news/1	..	http://site.ru/news
http://site.ru/news/1	../../contacts	http://site.ru/contacts

Использовать относительные адреса для навигации по сайту не рекомендуется. Однако относительные адреса бывают полезны, например, во внешних CSS-файлах.

Код в HTML

```
<!DOCTYPE html>
<html>
```

```
  <head>
    <meta charset="utf-8">
    <title>Относительные адреса</title>
  </head>
  <body>
    <h1>HTML Academy</h1>

    <p>Адрес текущей страницы
<code>https://npoed.htmlacademy.ru/htmlcss102/course/4/run/4</code>. Куда же ведут
ссылки с относительными адресами в списке ниже?</p>

    <ul>
      <li><a href="3">Предыдущее задание</a></li>
      <li><a href="..">Список заданий раздела</a></li>
      <li><a href="../../1">Раздел «Знакомство»</a></li>
    </ul>
  </body>
</html>
```

Код в CSS

```
code {
  padding: 3px;
  border: 1px solid #e1e1e8;
  background-color: #f7f7f9;
  color: #dd1144;
}
```

Задание 5.

- Задайте адрес ссылки `/assets/course4/file.rtf`,
- затем щёлкните по ней, чтобы файл скачался, либо открылся в мини-браузере.

Теоретический материал

Ссылка на файл

Вы узнали, чем отличаются абсолютные адреса от относительных. Теперь потренируемся ставить ссылки на различные ресурсы.

Ссылка может указывать на любую веб-страницу, на любой файл. Если щёлкнуть по ссылке, ведущей на файл, то браузер предложит его скачать.

Однако, если браузер умеет обрабатывать файлы этого типа, то содержимое файла откроется прямо в браузере. Чаще всего так происходит с изображениями. В последнее время браузеры научились открывать `.pdf` файлы и многие другие.

В этом задании вы создадите ссылку на простой текстовый файл в формате `.rtf`, который расположен на нашем сервере, и скачаете его.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ссылка на файл</title>
  </head>
```

```
  <body>
    <h1>Скачиваем файлы</h1>
    <p><a href="">Скачать простой текстовый файл</a></p>
  </body>
</html>
```

Задание 6.

- Задайте адрес ссылки `/assets/course4/img1.jpg`,
- затем щёлкните по ней, чтобы изображение открылось в мини-браузере.

Теоретический материал

Ссылка на изображение

Сделаем то же самое, но с изображением.
Пропишите ссылку на изображение, расположенное на нашем сервере, а затем откройте его.
Как известно, интернетом правят котики. И мы не будем отходить от традиции.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ссылка на изображение</title>
  </head>
  <body>
    <h1>Кекс в новой роли!</h1>
    <p><a href="">Знакомьтесь, Дон Кексоне!</a></p>
  </body>
</html>
```

Задание 7.

- Ссылке «История» задайте адрес `#history` и щёлкните по ней.
- Затем ссылке «Структура URL» задайте адрес `#structure` и тоже щёлкните по ней.
- Если при щелчке по ссылке с якорем перезагружается страница, нажимайте кнопку .

Теоретический материал

Ссылка с якорем

Ссылки с якорем обычно используются для создания навигации внутри страницы.
Например, оглавления в начале страницы с большой статьёй.
Ссылка с якорем содержит символ `#`, после которого идёт идентификатор.
Идентификатор создаётся с помощью атрибута `id`, который может быть задан у любого тега.
Можно задать адрес, состоящий из одного якоря, например:

```
<a href="#glava1">Глава 1</a>
```

При щелчке на такую ссылку браузер найдёт на странице элемент с атрибутом `id` со значением `glava1` и прокрутит окно страницы к нему. То есть перезагрузки страницы не произойдёт.
Якорь можно использовать и в абсолютных адресах, тогда после перехода на нужную страницу произойдёт прокрутка к заданной части этой страницы.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Ссылка с якорем</title>
   </head>
   <body>
      <h1>URL</h1>

      <h2 id="contents">Оглавление</h2>
      <ol>
         <li><a href="">История</a></li>
         <li><a href="">Структура URL</a></li>
         <li><a href="">Кодирование URL</a></li>
      </ol>

      <h2 id="history">История</h2>
      <p>URL был изобретён Тимом Бернерсом-Ли в 1990 году в стенах Европейского совета по ядерным исследованиям в Женеве, Швейцария. URL стал фундаментальной инновацией в Интернете.</p>
      <p>Изначально URL предназначался для обозначения мест расположения ресурсов (чаще всего файлов) во Всемирной паутине. Сейчас URL применяется для обозначения адресов почти всех ресурсов Интернета. Стандарт URL закреплён в документе RFC 1738, прежняя версия была определена в RFC 1630.</p>
      <p>Сейчас URL позиционируется как часть более общей системы идентификации ресурсов URI, сам термин URL постепенно уступает место более широкому термину URI. Стандарт URL регулируется организацией IETF и её подразделениями.</p>
      <a href="#contents">К оглавлению</a>

      <h2 id="structure">Структура URL</h2>
      <p>Изначально локатор URL был разработан как система для максимально естественного указания на местонахождения ресурсов в сети. Локатор должен был быть легко расширяемым и использовать лишь ограниченный набор ASCII-символов (к примеру, пробел никогда не применяется в URL). В связи с этим, возникла следующая традиционная форма записи URL:</p>

<p>&lt;схема&gt;://&lt;логин&gt;:&lt;пароль&gt;@&lt;хост&gt;:&lt;порт&gt;/&lt;URL-путь&gt;?&lt;параметры&gt;#&lt;якорь&gt;</p>
      <a href="#contents">К оглавлению</a>

      <h2 id="encoding">Кодирование URL</h2>
      <p>Появление адресов URL стало существенным нововведением в Интернете. Однако с момента его изобретения и по сей день стандарт URL обладает серьёзным недостатком — в нём можно использовать только ограниченный набор символов, даже меньший, нежели в ASCII: латинские буквы, цифры и лишь некоторые знаки препинания. Если мы захотим использовать в URL символы кириллицы, или иероглифы, или, скажем,
```

специфические символы французского языка, то нужные нам символы должны быть перекодированы особым образом.</p>
        К оглавлению
    </body>
</html>

Код в CSS

```
body {
    max-width: 350px;
    font-size: 16px;
    line-height: 24px;
}
```

Задание 8.

Добавьте всплывающие подсказки к двум ссылкам.

Теоретический материал

Всплывающая подсказка

Для того, чтобы добавить ссылке всплывающую подсказку, надо использовать атрибут `title`. Например:

```
<a title="Подсказка" href="#">
```

Подсказка появится, когда курсор задержится над ссылкой некоторое время. Подсказки помогают разъяснить назначение непонятных ссылок, а также ссылок-изображений.

Код в HTML

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Всплывающая подсказка</title>
    </head>
    <body>
        <h1>URL</h1>
        <h2 id="contents">Оглавление</h2>

        <ol>
            <li><a href="#history">История</a></li>
            <li><a href="#structure">Структура URL</a></li>
        </ol>
    </body>
</html>
```

Задание 9.

- Добавьте на страницу тег `<img>`
- с атрибутом `src` со значением `/assets/course4/img1.jpg`.

Теоретический материал

Добавим изображение

Чтобы добавить на страницу изображение, нужно использовать одиночный тег `<img>` с атрибутом `src`, в котором указан адрес картинки. Например:

```
<img src="logo.png">
```
В этом задании мы добавим на страницу изображение кота из шестого задания.
Кстати, самыми распространёнными форматами изображений в сети
являются JPEG и PNG.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Добавим изображение</title>
  </head>
  <body>
    <h1>Дон Кексоне</h1>
  </body>
</html>
```

Задание 10.

Последовательно задайте разные размеры для изображения Кекса:

- Сначала ширину `100` процентов.
- Затем ширину `200` пикселей.
- Затем ширину и высоту `200` пикселей.

Теоретический материал

Размеры изображения

Чтобы управлять шириной или высотой изображения, нужно использовать атрибуты `width` и `height`. Пример:

```
<img width="100" src="...">
```

В примере изображению задана ширина `100px`. Обратите внимание, что в атрибуте `width` после цифры нет `px`. Если вы задаёте размер картинки в пикселях, то используйте просто цифры. Добавлять `px` *не* нужно, таков стандарт.

Во втором примере изображению задана относительная ширина, `50` процентов:

```
<img width="50%" src="...">
```

Высоту в процентах обычно не задают.
Если задать только один из размеров, ширину или высоту, то вторую размерность браузер вычислит самостоятельно исходя из пропорций изображения.
Если же задать и ширину, и высоту для картинки:

```
<img width="100" height="100" src="...">
```

То браузер может нарушить пропорции исходного изображения.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Размеры изображения</title>
```

```
  </head>
  <body>
    <h1>Дон Кексоне</h1>
    <img src="/assets/course4/img1.jpg">
  </body>
</html>
```

Задание 11.

Задайте изображению альтернативный текст `Рыжий кот`,

а затем измените значение `src` на любое другое, чтобы браузер не мог загрузить картинку.

Теоретический материал

Альтернативный текст

Если у пользователя отключены изображения или их невозможно загрузить, то в браузере отображается альтернативный текст. Например, если меню сделано с помощью изображений, то альтернативный текст поможет понять, куда ведёт каждый пункт.
В общем, задавать альтернативный текст хорошо.

Альтернативный текст изображения задаётся с помощью атрибута `alt`. Пример:

```
<img src="cat.png" alt="Кот в полном расцвете сил">
```

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Альтернативный текст</title>
  </head>
  <body>
    <h1>Дон Кексоне</h1>
    <img src="/assets/course4/img1.jpg" width="400">
  </body>
</html>
```

Задание 12.

- Добавьте ссылку с маленького изображения на большое
  с адресом `/assets/course4/img1.jpg`.
- Затем щёлкните по этой ссылке.

Теоретический материал

Изображение-ссылка

Ссылки можно делать не только с помощью текста, но и с помощью изображений. Для этого нужно обернуть тег `<img>` в тег `<a>`. Например:

```
<a href="http://keksby.ru">
  <img src="cat.png" alt="Кекс">
</a>
```

Часто ссылки-изображения используются в галереях, когда с уменьшенной версии изображения ставится ссылка на полноразмерное изображение.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Изображение-ссылка</title>
   </head>
   <body>
      <h1>Дон Кексоне</h1>
      <img src="/assets/course4/img1_small.jpg">
   </body>
</html>
```

Задание 12.

- Задайте адреса всех шести изображений.

Теоретический материал

Фоторепортаж

В конце курса на этот раз простое развлекательное задание. Вы поможете завершить небольшой фоторепортаж. Для этого будет нужно задать правильные адреса изображений.

Адрес первого изображения: `/assets/course4/photo1.jpg`.

Адрес второго изображения: `/assets/course4/photo2.jpg` и так далее до цифры `6`

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Фоторепортаж</title>
   </head>
   <body>
      <h1>День кота</h1>

      <p>В то утро Кексик проснулся в своей сумке-гнезде очень рано. День обещал быть насыщенным.</p>
      <img alt="Пробуждение">

      <p>В связи с чем он решил позаниматься на своей вешалке-турнике.</p>
      <img alt="Зарядка">

      <p>Занятия утомили кота и он решил слегка отдохнуть, посмотреть телевизор.</p>
      <img alt="Телевизор">

      <p>Восстановив силы, кот занялся привычными делами по наведению беспорядка, но был пойман.</p>
      <img alt="Пойман раз">

      <p>Ближе к вечеру подвернулась отличная возможность отведать сметаны, но и в этот раз его поймали на месте преступления.</p>
```

```
    <img alt="Пойман два">

    <p>Решив, что на сегодня свершений достаточно, Кексик с чистой совестью отправился спать.</p>
    <img alt="Сон">
  </body>
</html>
```

Код в CSS

```
img {
  max-width: 80%;
  border-radius: 10px;
  box-shadow: 0 0 3px #999;
}
```

Лабораторная работа 8-9
Работа с текстом.

Задание1

Разметьте с помощью тега `p` ещё три предложения:

1. *«Абзац служит для ... единиц изложения.»*
2. *«Выделение фразы в ... смысловой акцент.»*
3. *«Для выделения абзаца ... абзацный отступ.»*

Теоретический материал

Абзацы

В разделе «Структура HTML-документа» вы познакомились с тегами, необходимыми для создания простейшей HTML-страницы, и с некоторыми служебными тегами, которые не отображаются в браузере.

В этом разделе мы будем изучать теги для логической разметки текста. Использовать их можно только внутри тега `<body>`.

Начнём с простейшего тега `<p>`, с помощью которого создаются абзацы. По умолчанию абзацы начинаются с новой строки и имеют вертикальные отступы, которыми можно управлять с помощью стилей.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Абзацы</title>
  </head>
  <body>
    <p>Абзац — отрезок письменной речи, состоящий из нескольких предложений.</p>

    Абзац служит для группировки однородных единиц изложения.

    Выделение фразы в особый абзац усиливает падающий на неё смысловой акцент.

    Для выделения абзаца его обычно печатают с красной строки или делают абзацный отступ.
```

Вот такие они, абзацы.
</body>
</html>

Задание2

Оберните в тег `h1` текст *«1 Заголовки»*,

в тег `h2` текст *«1.1 Рекомендации по использованию»*,

в теги `h3` тексты *«1.1.1 Использование H1»* и *«1.1.2 Использование H2-H6»*.

Теоретический материал

Заголовки и подзаголовки

Для создания структуры больших текстов обычно используются заголовки. В текстовых редакторах есть возможность выделить часть текста, найти пункт «Заголовок» нужного уровня в меню, и применить его.

В языке HTML для выделения заголовков предусмотрено целое семейство тегов: от `<h1>` до `<h6>`. Тег `<h1>` обозначает самый важный заголовок (заголовок верхнего уровня), а тег `<h6>` обозначает подзаголовок самого нижнего уровня.

На практике редко встречаются тексты, в которых встречаются подзаголовки ниже третьего уровня. Поэтому самыми часто используемыми тегами заголовков являются: `<h1>`, `<h2>` и `<h3>`.

Стоит отметить, что поисковые системы придают особое значение заголовкам, поэтому необходимо учиться правильно их использовать.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Заголовки и подзаголовки</title>
  </head>
  <body>
    1 Заголовки
    <p>Заголовки являются одним из важнейших инструментов для структурирования текста.</p>

    1.1 Рекомендации по использованию
    <p>Вот некоторые рекомендации по использованию заголовков в HTML-странице.</p>

    1.1.1 Использование H1
    <p>Не рекомендуется, чтобы на одной странице содержалось несколько заголовков верхнего уровня.</p>

    1.1.2 Использование H2-H6
    <p>При использовании подзаголовков не рекомендуется пропускать уровни заголовков, то есть после заголовка H1 должен идти подзаголовок H2 и только потом подзаголовок H3.</p>
  </body>
</html>
```

Задание3

- добавить в список изученных тегов ещё минимум три пункта.

Теоретический материал

Неупорядоченный список

Списки часто используются в различных документах. Иногда, чтобы сделать список, пользователь просто нумерует строчки текста. Такой подход не является хорошим, так как в документе отсутствует логическая сущность «список».

В HTML существует семейство тегов для создания списков: неупорядоченных, упорядоченных и списков определений. В последующих заданиях мы будем тренироваться работать с ними.

Неупорядоченные (или маркированные) списки создаются с помощью тега `<ul>`, который может содержать внутри себя теги `<li>`, обозначающие «элемент списка».

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Неупорядоченный список</title>
  </head>
  <body>
    <h1>Список изученных тегов</h1>
    <ul>
      <li>html</li>
      <li>head</li>
      <li>body</li>
      <!-- добавьте пункты сюда -->
    </ul>
    <p>Неупорядоченный список используется для простого перечисления объектов, когда порядок неважен.</p>
  </body>
</html>
```

Задание4

- Добавьте в список заголовков ещё минимум три пункта,
- Затем измените начало нумерации списка с помощью атрибута `start` так, чтобы она начиналась с числа `2` или больше.

Теоретический материал

Упорядоченный список

Упорядоченный список создаётся с помощью тега `<ol>`, который может содержать внутри себя теги `<li>`.

Если элементы неупорядоченного списка по умолчанию отмечаются маркерами, то элементы упорядоченного списка — нумеруются.

Для упорядоченного списка можно задать атрибут `start`, который изменяет начало нумерации. Например, код:

```
<ol start="3">
  <li>раз</li>
  <li>два</li>
</ol>
```
Приведёт к такому результату:

3. раз
4. два

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Упорядоченный список</title>
   </head>
   <body>
      <h1>Какие бывают заголовки?</h1>
      <p>Теги заголовков и подзаголовков в порядке убывания значимости:</p>
      <ol>
         <li>h1</li>
         <li>h2</li>
         <li>h3</li>
         <!-- добавьте пункты сюда -->
      </ol>
      <p>Упорядоченный список используется для перечисления объектов или действий, когда важен порядок.</p>
   </body>
</html>
```

Задание5

- Добавьте ещё два пункта во вложенный список «Теги заголовков».
- Добавьте в пункт «Теги списков» неупорядоченный вложенный список из двух пунктов.

Теоретический материал

Многоуровневый список

Создать многоуровневый список достаточно просто.

Сначала нужно создать список первого уровня, а затем внутрь любого элемента этого списка, между тегами `<li>` и `</li>`, добавить список второго уровня. При этом необходимо аккуратно закрывать все теги.

Пример правильного кода:

```
<ul>
  <li>1
    <ul>
      <li>1.1</li>
      <li>1.2</li>
    </ul>
  </li>
  <li>2</li>
</ul>
```

Пример кода с ошибкой:

```
<ul>
```

```
 <li>1</li>
 <ul>
  <li>1.1</li>
  <li>1.2</li>
 </ul>
 <li>2</li>
</ul>
```

В примере с ошибкой вложенный список вставлен не внутрь элемента списка, а между элементами, что недопустимо.
Количество уровней в списках не ограничено. В многоуровневом списке можно использовать как упорядоченные, так и неупорядоченные списки.
В этом задании мы потренируемся работать с многоуровневыми списками.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Многоуровневый список</title>
  </head>
  <body>
    <h1>Разделы курса</h1>
    <ol>
      <li>
        Теги заголовков
        <ol>
          <li>h1</li>
          <li>h2</li>
          <!-- добавьте элементы вложенного списка для первой цели сюда -->
        </ol>
      </li>
      <li>
        Теги списков
        <!-- добавьте вложенный список для второй цели сюда -->
      </li>
      <li>Теги форм</li>
    </ol>
  </body>
</html>
```

Задание6

- Создайте четырёхуровневый упорядоченный список. Достаточно, чтобы на каждом уровне было по одному пункту.

Теоретический материал

Ещё более многоуровневый

Хорошо. Вы создали двухуровневый список. Теперь задание посложнее.
В этом задании вам нужно будет создать четырёхуровневый список, наподобие этого:

1. Разметка
    1. Основы HTML
        1. HTML-теги

1. парные
2. одиночные

2. Основы CSS
    1. Селекторы
        1. по типу
        2. по классу
        3. вложенные
    3. Стиль кодирования
2. Работа с фотошопом
3. Построение сеток
4. Декоративные элементы
5. Введение в JavaScript
6. Прогрессивное улучшение

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ещё более многоуровневый</title>
  </head>
  <body>
    <h1>Невероятно многоуровневый список</h1>
    <ol>
      <li>Первый элемент</li>
      <li>Второй элемент</li>
    </ol>
  </body>
</html>
```

Задание7

В существующий список определений добавьте:

- Четвёртый пункт с термином `ul` и определением `Unordered List, неупорядоченный список`.
- Пятый пункт с термином `li` и определением `List Item, элемент списка`.

Теоретический материал

Список определений

Список определений создаётся с помощью трёх тегов:

1. `<dl>` обозначает сам список определений;
2. `<dt>` обозначает термин;
3. `<dd>` обозначает определение термина.

Теги `<dt>` и `<dd>` пишутся парами внутри `<dl>`.

Например:

```
<dl>
 <dt>Термин</dt>
 <dd>Определение</dd>

 <dt>Второй термин</dt>
```

```
  <dd>И его определение</dd>

  <dt>Кошка</dt>
  <dd>Шерстяное изделие развлекательного характера</dd>
</dl>
```

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Список определений</title>
   </head>
   <body>
      <h1>Расшифровка тегов списков</h1>
      <dl>
         <dt>dl</dt>
         <dd>Definition List, сам список определений</dd>

         <dt>dt</dt>
         <dd>Definition Term, термин</dd>

         <dt>dd</dt>
         <dd>Definition Definition, определение термина</dd>
      </dl>
   </body>
</html>
```

Задание8

- Выделите вводный абзац: заключите внутрь тега `b` весь текст внутри первого абзаца.
- Выделите важные детали: заключите внутрь тега `strong` слово «профессионала» во втором абзаце
- и фразу «настоящим ИТ-специалистом» в третьем абзаце.

Теоретический материал

Важность. Теги strong и b

Ещё раз отметим, что этот раздел посвящён **логической** разметке текста, поэтому уделяется особое внимание смыслу элементов, их предназначению, а не визуальному форматированию.

В предыдущих заданиях вы познакомились с элементами, которые предназначены для разметки крупных блоков текста: заголовков, абзацев и списков. В этом и последующих заданиях мы познакомимся с элементами, предназначенными для разметки небольших фраз и отдельных слов.

Первые два тега предназначены, чтобы указать на важность слова или фразы.

Тег `<strong>` определяет **важность** отмеченного текста.

Тег `<b>` предназначен для выделения текста без придания ему особой важности.

Лучше всего отличия этих тегов будут заметны людям, которые используют специальные настройки ОС, в частности, слепым и слабовидящим. Когда они включают функцию

чтения текста, то «говорилка» будет интонацией выделять слова с тегом . То же самое касается и тегов и <i>. Тег «говорилка» будет выделять интонацией. Визуально оба тега одинаковы, они выделяют текст полужирным.
Отметим, что новый смысл тегу <b> придали в HTML5. Раньше это был тег, который просто делает текст полужирным. То есть он был предназначен только для визуального форматирования.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Теги strong и b</title>
  </head>
  <body>
    <h1>Что не так с онлайн-курсами и как это исправить: опыт HTML Academy</h1>

    <p>Мы в HTML Academy постоянно работаем над повышением эффективности наших образовательных программ и курсов (ради этого мы ставим опыты на студентах и внедряем геймификацию).</p>

    <p>Подобные изыскания привели нас к пониманию того, что для подготовки профессионала, который мог бы работать в области веб-технологий (хотя бы на позиции начального уровня) недостаточно одних курсов, состоящих из видеолекций и последующих заданий. Освоить весь объём контента, необходимого для выхода на определённый уровень знаний, в подобном режиме просто невозможно.</p>

    <p>Именно поэтому мы решили создать курс, прохождение которого позволяло бы людям не просто получить базовые знания вёрстки, но стать настоящим ИТ-специалистом. В сегодняшнем материале мы расскажем о том, что в итоге из всего этого получилось.</p>

    <p>Прочитать <a href="http://habrahabr.ru/company/htmlacademy/blog/252843/" target="_blank" rel="nofollow">статью целиком</a> вы можете на Хабре.</p>
  </body>
</html>
```

Задание9

- Расставьте акценты: заключите внутрь тега em слово «профессию» в первом абзаце
- и слово «специалистом» в том же абзаце.
- Выделите с помощью тега i термин «офлайн» во втором абзаце.

Теоретический материал

Акцентируем внимание. Теги em и i

Следующие два тега предназначены для акцентирования внимания на слово или фразу. Тег определяет текст, на который сделан *особый акцент*, меняющий смысл предложения.
Например, если мы хотим подчеркнуть, что Кекс не любит *питаться* укропом (он больше за тунца), а любит только *гонять его по полу*, то разметим текст так:

```
Инструктор Кекс любит <em>играть</em> с укропом.
```

Тег <i> обозначает текст, который отличается от окружающего текста, но не является более важным. Обычно так выделяют *названия*, *термины*, *иностранные слова*.
Например, если мы хотим указать, что *инспектор* — это какой-то специальный термин, то разметим текст так:

```
Обычно Кекс пользовался <i>инспектором</i> браузера для поиска ошибок.
```

Визуально оба тега одинаковы, они выделяют текст курсивом.
Новый смысл тегу <i> придали в HTML5. Раньше это был просто тег для выделения текста курсивом.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Теги em и i</title>
  </head>
  <body>
    <h1>Проблема: что не так с онлайн-курсами</h1>

    <p>Ученикам, которые хотели бы не просто разобраться с той или иной технологией, а ищут возможность получить новую профессию или стать более востребованным специалистом, нужно предоставить возможность каким-то образом подтвердить уровень своих знаний — без работы с наставниками и выполнения выпускных проектов с последующим получением сертификата этого не сделать.</p>

    <p>Все вышеперечисленное прекрасно существует в режиме офлайн, но можно ли перенести обучение такого формата в интернет?</p>

    <p>Прочитать <a href="http://habrahabr.ru/company/htmlacademy/blog/252843/" target="_blank" rel="nofollow">статью целиком</a> вы можете на Хабре.</p>
  </body>
</html>
```

Задание10

- С помощью тега br добавьте три переноса строки в первый куплет,
- два переноса во второй куплет,
- три переноса в третий.
- Добавьте разделители между абзацами куплетов с помощью тега hr.

Теоретический материал

Переносы и разделители. Теги br и hr
Иногда возникает необходимость вставить в текст перенос строки, не создавая при этом абзац. Например, при разметке стихов или текстов песен.
Для этого в HTML предусмотрен одиночный тег
.
Иногда этот тег используется для разбиения текста на «как бы абзацы», что является плохим подходом. Используйте для разметки абзацев тег <p>.
Одиночный тег <hr> используется для того, чтобы создать горизонтальную линию-разделитель. На внешний вид этой линии можно влиять с помощью атрибутов, но правильней делать это с помощью CSS (это будет изучаться в последующих разделах).

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Переносы и разделители</title>
  </head>
  <body>
    <h1>Чайф. С войны</h1>

    <p>
      В твоём парадном темно,
      Резкий запах привычно бьёт в нос.
      Твой дом был под самой крышей —
      В нём немного ближе до звёзд.
    </p>

    <p>
      Ты шёл не спеша, возвращаясь с войны
      Со сладким чувством победы,
      С горьким чувством вины.
    </p>

    <p>
      Вот твой дом, но в двери
      Уже новый замок.
      Здесь ждали тебя так долго,
      Но ты вернуться не мог.
    </p>
  </body>
</html>
```

Задание11

1. фразы *«Мастер и Маргарита»*,
2. **Цель 2**фразы *«Никогда ничего не просите ... Сами предложат и сами всё дадут!»*,
3. **Цель 3**и фразы *«Hasta la vista, baby»*.

Теоретический материал

Цитаты

В HTML существует несколько тегов для обозначения цитат:

- `<blockquote>` предназначен для выделения длинных цитат, которые могут состоять из нескольких абзацев. Тег выделяет цитату как отдельный блок текста с отступами.
- `<q>` предназначен для выделения коротких цитат в тексте предложения. Текст внутри этого тега автоматически обрамляется кавычками.
- `<cite>` используется для того, чтобы выделить источник цитаты, название произведения, но не автора цитаты.

Код в HTML

```
<!DOCTYPE html>
<html>
```

```
<head>
    <meta charset="utf-8">
    <title>Цитаты</title>
</head>
<body>
    <h1>Цитаты великих</h1>

    <p>В книге Мастер и Маргарита есть такая цитата:</p>

    Никогда ничего не просите! Никогда и ничего, и в особенности у тех, кто сильнее вас. Сами предложат и сами всё дадут!

    <p>Известное выражение Hasta la vista, baby принадлежит Терминатору.</p>

    <p>Кекс предпочитает поговорку Без труда не выловишь и рыбку из пруда! и повторяет её постоянно.</p>
</body>
</html>
```

Задание12

1. Приведите первую формулу к виду H_2SO_4,
2. **Цель 2**вторую формулу к виду $sin^2x+cos^2x=1$.
3. **Цель 3**Третью формулу приведите к подходящему виду самостоятельно.

Теоретический материал

Верхние и нижние индексы [12/17]

Следующие два тега обычно используются не для выделения слов, а для выделения отдельных символов. Их используют для указания единиц измерения или для написания простых формул.

Например: $20м^2$, H_2O, $X^3+X^2=1$

Тег `<sup>` отображает текст в виде верхнего индекса.

Тег `<sub>` отображает текст в виде нижнего индекса.

Стоит отметить, что эти теги являются чисто презентационными и не имеют собственной семантики.

Эти теги можно использовать внутри друг друга для создания более сложных формул.

Если вам нужно вставить очень сложную формулу в HTML-документ, лучше воспользоваться специальным языком разметки MathML.

Код в HTML

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Верхние и нижние индексы</title>
    </head>
    <body>
        <h1>Формулы</h1>
        <ul>
            <li>H2SO4</li>
            <li>sin2x+cos2x=1</li>
            <li>C2H5OH</li>
            <li>e-x2</li>
```

```
    </ul>
  </body>
</html>
```

Задание13

- Пометьте в списке ещё одну ошибку как исправленную с помощью `del`,
- Цель 2и ещё одну как новую с помощью `ins`.

Теоретический материал

Помечаем изменения. Теги del и ins

Любой документ на протяжении своей «жизни» может изменяться. С распространением динамических веб-приложений вносить изменения в HTML-документы стало проще простого.

Иногда возникает вопрос: а что же именно было изменено в документе, что было добавлено, а что удалено?

Как раз для описания изменений предназначены теги `<del>` и `<ins>`.

`<del>` выделяет текст, который был удалён в новой версии документа.

`<ins>` выделяет текст, который был добавлен в новой версии документа.

Оба тега имеют атрибут `datetime`, в котором можно указать дату и время, когда была внесена та или иная правка.

Простейшим примером применения этих тегов может служить список ошибок. Когда ошибка исправлена, её помечают тегом `<del>`, если найдена новая ошибка,

то её добавляют в список и помечают тегом `<ins>`.

Атрибут `datetime` предназначен не для людей, а для компьютеров, поэтому дату и время там пишут в стандартизованном формате. При такой разметке программам легче разбирать документы и анализировать, когда произошли те или иные изменения.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Теги del и ins</title>
  </head>
  <body>
    <h1>Ошибки, 06.05.2015</h1>
    <ul>
      <li><del datetime="2015-05-07">Пункты меню слипаются.</del></li>
      <li>Неверный размер шрифта в заголовке.</li>
      <li>Логотип не обновлён.</li>
      <li><ins datetime="2015-05-08">Ошибки в текстах</ins>.</li>
      <li>Макет разваливается на минимальной ширине.</li>
    </ul>
  </body>
</html>
```

Задание14

Заключите в тег `pre` весь текст внутри тега `body`.

Преформатированный текст
Наверное, вы уже заметили, как отличается отображение кода в HTML-редакторе и в мини-браузере.
Вы можете ставить сколько угодно пробелов в HTML-коде, но браузер отобразит их как один. Вы также можете ставить сколько угодно переносов строки в HTML-коде, а в браузере переноса не будет, если только не использовать специальные теги, например `<p>` или `<br>`.

Изменить это поведение браузера можно с помощью тега `<pre>`, который обозначает «предварительно отформатированный текст». Браузер сохраняет и отображает все пробелы и переносы, которые есть внутри тега `<pre>`.
Наиболее часто этот тег используется при отображении примеров кода.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Преформатированный текст</title>
   </head>
   <body>

 /\ \  __/\ \       /\ \
 \ \ \/\ \ \ \    __\ \ \_____
  \ \ \ \ \ \ \  /'__`\ \ '__`\
   \ \ \_/ \_\ \/\  __/\ \ \_\ \
    \ `\___x___/\ \____\\ \_,__/
     \/__//__/  \/____/ \/___/

   </body>
</html>
```

Задание15

Выделите три любых фрагмента текста с помощью тега `mark`.
Просто выделенный текст
В HTML5 появился новый тег `<mark>`, который обозначает выделенный текст.
Иногда при работе с объёмными текстами мы используем маркер, чтобы выделять ключевые слова, идеи или что-то другое на что стоит обратить внимание. Такое же назначение и у тега `<mark>`.

В современных браузерах текст внутри `<mark>` подсвечивается жёлтым фоном.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Выделенный текст</title>
```

```
  </head>
  <body>
    <h1>Как это работает: онлайн-интенсивы</h1>

    <p>Минусы привычных онлайн-курсов понятны, но что если сделать этот инструмент лишь одной из ступеней обучения, дополнив программу занятиями с наставниками и разработкой проектов, приближенным к «боевым»? Мы решили пойти этим путём и создали учебную программу базовых интенсивов. Она включает пять основных элементов:</p>

    <ul>
      <li>Онлайн-курсы для знакомства с веб-технологиями и освоения базовых навыков работы с ними.</li>
      <li>Вебинары, на которых слушатели получают больше продвинутой теоретической информации, чем в ходе онлайн-курсов.</li>
      <li>Личное общение с наставником — опытные специалисты, сотрудничающие с HTML Academy, связываются с учениками по скайпу и в режиме 1 на 1 помогают им осваивать программу курса и проверяют домашние задания (индивидуальный подход гораздо эффективнее групповых занятий в офлайне).</li>
      <li>Сквозные проекты — в ходе обучения студенты работают над реальными проектами и шаг за шагом верстают настоящий сайт, который ничем не отличается от обычных корпоративных веб-ресурсов.</li>
      <li>Сертификация — как уже было сказано выше, люди хотят получить «плюсик в резюме», поэтому им нужно дать возможность защитить свой проект на соответствие ряду критериев и получить реальный сертификат о прохождении обучения.</li>
    </ul>

    <p>Прочитать <a href="http://habrahabr.ru/company/htmlacademy/blog/252843/" target="_blank" rel="nofollow">статью целиком</a> вы можете на Хабре.</p>
  </body>
</html>
```

Задание16

Образец

Пшенная каша с тыквой

1. Пшено, 1 стакан
2. Вода H_2O или AquaLife®, 2 стакана
3. Молоко, ~~1,5 стакана~~ 2 стакана
4. Масло сливочное, 3 ст.л.
5. Тыква, около 300г
6. Соль, по вкусу
7. Сахар демерара, посыпать сверху

Примечания:
Пшено необходимо перебрать
Тыкву нарезать кубиками 1x1 см
Кашу перемешивать не надо

Испытание: разметка статьи
Чтобы пройти испытание, вам необходимо разметить текст в HTML-редакторе точно так же, как и в образце.
При создании образца были использованы только те HTML-теги, которые изучались в данном курсе. CSS-стили не использовались, так что изменять код CSS-редактора нельзя.
Если отображение в мини-браузере сильно отличается от образца, то используйте вкладку «Различия», в которой отображается то, как наш сервер видит ваш HTML-код.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Рецепт каши</title>
  </head>
  <body>
    Пшенная каша с тыквой

      Пшено, 1 стакан
      Вода H2O или AquaLife®, 2 стакана
      Молоко, 1,5 стакана 2 стакана
      Масло сливочное, 3 ст.л.
      Тыква, около 300г
      Соль, по вкусу
      Сахар демерара, посыпать сверху

    Примечания:
    Пшено необходимо перебрать
    Тыкву нарезать кубиками 1x1 см
    Кашу перемешивать не надо

  </body>
</html>
```

Лабораторная работа 10-11
Создание таблицы

Простейшая таблица
Из всех объектов, которые используются для разметки текста в интернете, таблицы являются самыми сложными для новичков. Действительно, табличные данные приходится публиковать достаточно часто. В отличие от списков, абзацев, заголовков, изображений с таблицами всегда возникает море проблем.
В этом разделе мы узнаем, как с помощью HTML описывать таблицы, научимся делать простые и достаточно сложные таблицы. И, самое главное, научимся аккуратно

оформлять таблицы с помощью CSS. Вы увидите, как на самом деле легко и просто работать с таблицами.
Начнём с самого простого. Уберите комментарий в коде редактора и посмотрите на простейшую таблицу из четырёх строк и двух колонок.

Задание 1. Уберите комментарий в коде HTML-редактора.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Простейшая таблица</title>
  </head>
  <body>
    <h1>Распределение браузеров</h1>
<!--
    <table border="1">
      <tr>
        <td>Браузер</td>
        <td>Посещения</td>
      </tr>
      <tr>
        <td>Mozilla Firefox</td>
        <td>163</td>
      </tr>
      <tr>
        <td>Google Chrome</td>
        <td>78</td>
      </tr>
      <tr>
        <td>Safari</td>
        <td>35</td>
      </tr>
    </table>
 -->
  </body>
</html>
```

Добавляем строки

Простейшая таблица описывается с помощью трёх тегов:

1. `<table>` обозначает таблицу.
2. `<tr>` расшифровывается как *«table row»*, обозначает строку таблицы.
3. `<td>` расшифровывается как *«table data»*, обозначает ячейку внутри строки таблицы.

Теги `<td>` располагаются внутри тегов `<tr>`, а те, в свою очередь, внутри `<table>`. Почти всё текстовое содержимое таблицы размещается внутри тегов `<td>`.

В простейшей таблице в каждой строке должно быть одинаковое количество ячеек, то есть внутри всех `<tr>` должно быть одинаковое количество `<td>`.

Потренируйтесь добавлять строки в таблицу.

Задание 2.

- Добавьте в таблицу пятую строку со статистикой для браузера `Opera`,
- Шестую строку со статистикой для браузера `Internet Explorer`.
- Количество посещений можете указать любое.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Добавляем строки</title>
  </head>
  <body>
    <h1>Распределение браузеров</h1>

    <table border="1">
      <tr>
        <td>Браузер</td>
        <td>Посещения</td>
      </tr>
      <tr>
        <td>Mozilla Firefox</td>
        <td>163</td>
      </tr>
      <tr>
        <td>Google Chrome</td>
        <td>78</td>
      </tr>
      <tr>
        <td>Safari</td>
        <td>35</td>
      </tr>
    </table>

  </body>
</html>
```

Добавляем столбцы

Со строками справились, теперь потренируемся добавлять в таблицу столбцы.

Для того, чтобы добавить столбец в таблицу, надо в каждую строку `<tr>` добавить по ячейке `<td>`.

В этом задании вам нужно будет в исходную таблицу добавить ещё два столбца. То есть в каждой строке должно быть по четыре ячейки.

Задание 3.

- Добавьте в таблицу третий столбец `Операционная система` и четвёртый столбец `Посещаемость в процентах`.

- Значения в первой строке каждого столбца должны быть как в задании, значения в остальных ячейках можете указать любые.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Добавляем столбцы</title>
  </head>
  <body>
    <h1>Распределение браузеров</h1>

    <table border="1">
      <tr>
        <td>Браузер</td>
        <td>Посещения</td>
      </tr>
      <tr>
        <td>Mozilla Firefox</td>
        <td>163</td>
      </tr>
      <tr>
        <td>Google Chrome</td>
        <td>78</td>
      </tr>
      <tr>
        <td>Safari</td>
        <td>35</td>
      </tr>
    </table>

  </body>
</html>
```

Задаём рамки с помощью CSS

Вы научились создавать простые таблицы, добавлять в них любое количество строк и столбцов. Теперь пришло время оформить эти таблицы.

Таблицы в предыдущих заданиях отображались с рамками по умолчанию. Такие рамки отображаются, если у тега `<table>` задан атрибут `border` с ненулевым значением.

Но с помощью атрибута `border` гибко управлять рамками не получается. С его помощью можно только изменять их толщину.

Поэтому мы будем учиться использовать CSS. С помощью CSS-свойства `border` можно задавать как внешние рамки таблицы, так и рамки каждой ячейки.

Потренируемся использовать CSS для задания рамок таблицы.

Задание 4.

- Измените значение атрибута `border` на `5` в HTML-редакторе.
- Затем удалите этот атрибут.
- В CSS-редакторе раскомментируйте стили для таблицы,
- затем раскомментируйте стили для ячейки.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Задаём рамки с помощью CSS</title>
   </head>
   <body>
      <h1>Распределение браузеров</h1>

      <table border="1">
         <tr>
            <td>Браузер</td>
            <td>Посещения</td>
         </tr>
         <tr>
            <td>Mozilla Firefox</td>
            <td>163</td>
         </tr>
         <tr>
            <td>Google Chrome</td>
            <td>78</td>
         </tr>
         <tr>
            <td>Safari</td>
            <td>35</td>
         </tr>
      </table>

   </body>
</html>
```

Код в CSS

```
/*
table {
   border: 3px solid black;
}
*/

/*
td {
   border: 3px solid lightgray;
}
*/
```

Улучшаем отображение рамок

Мы задали рамки таблицы с помощью CSS, но они не так хороши, как хотелось бы. По умолчанию браузер рисует рамки таблицы и рамки отдельных ячеек раздельно, это отлично видно на примере.

Чтобы избавиться от таких двойных рамок, используется CSS-свойство таблицы `border-collapse`. Вот так:

```
table {
  border-collapse: collapse;
}
```

Значение `collapse` убирает двойные рамки: схлопываются рамки соседних ячеек, а также рамки ячеек и внешняя рамка таблицы. При этом внешняя рамка таблицы может исчезнуть, и чтобы её вернуть, можно увеличить её ширину.

Задание 5.

- В CSS-редакторе задайте для таблицы свойство `border-collapse: collapse`, чтобы рамки ячеек схлопнулись.
- Затем в CSS-редакторе увеличьте ширину рамки *таблицы* до `6px`.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Улучшаем отображение рамок</title>
   </head>
   <body>
      <h1>Распределение браузеров</h1>

      <table>
         <tr>
            <td>Браузер</td>
            <td>Посещения</td>
         </tr>
         <tr>
            <td>Mozilla Firefox</td>
            <td>163</td>
         </tr>
         <tr>
            <td>Google Chrome</td>
            <td>78</td>
         </tr>
         <tr>
            <td>Safari</td>
            <td>35</td>
         </tr>
      </table>

   </body>
</html>
```

Код в CSS

```
table {
   border: 3px solid black;
}
```

```
td {
   border: 3px solid lightgray;
}
```

Горизонтальные и вертикальные рамки

Иногда требуется, чтобы рамки ячеек в таблице отображались не полностью. Например, чтобы отображалась только нижняя рамка ячеек, тогда таблица получается расчерченной по горизонтали. Аналогично, если отображать только боковые рамки ячеек, то таблица получается разбитой на столбцы.

Такие эффекты легко достигаются с помощью CSS. Для этого необходимо использовать не свойство `border`, которое задаёт рамки для всех сторон ячейки, а одно из свойств:

- `border-top`,
- `border-right`,
- `border-bottom`,
- `border-left`.

Эти свойства задают отображение только одной рамки ячейки: верхней, правой, нижней или левой соответственно.

В этом задании вы потренируетесь создавать горизонтально и вертикально расчерченные таблицы.

Задание 6.

- В CSS у *ячейки* удалите свойство `border`,
- затем раскомментируйте свойство `border-bottom`,
- затем удалите `border-bottom` и раскомментируйте `border-left`.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Горизонтальные и вертикальные рамки</title>
   </head>
   <body>
      <h1>Распределение браузеров</h1>

      <table>
         <tr>
            <td>Браузер</td>
            <td>Посещения</td>
            <td>Доля</td>
         </tr>
         <tr>
            <td>Mozilla 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>
```

Код в CSS

```
table {
    border-collapse: collapse;
    border: 3px solid black;
}

td {
    border: 2px solid lightgray;
    /* border-bottom: 2px solid lightgray; */
    /* border-left: 2px solid lightgray; */
}
```

Отступы внутри ячеек

Вы освоили простейшие приёмы для работы с рамками таблиц. Наша таблица уже смотрится аккуратно, но содержимое ячеек прилипает к рамкам. Если добавить отступы внутри ячеек, то информация будет восприниматься намного лучше.

Отступы внутри ячеек можно добавлять с помощью атрибута `cellpadding` тега `<table>`. Но лучше его не использовать, а задавать отступы с помощью CSS.

CSS-свойство `padding` задаёт «внутренние отступы элемента» со всех сторон. Можно задавать отступы для каждой из сторон отдельно, используя свойства:

- `padding-top`,
- `padding-right`,
- `padding-bottom`,
- `padding-left`.

Например, чтобы задать у ячеек все отступы в `10` пикселей, а отступ слева в `20` пикселей, нужно написать такой CSS-код:

```
td {
  padding: 10px;
  padding-left: 20px;
}
```

Задание 7.

- Сначала в HTML-редакторе измените значение атрибута `cellpadding` на `5`.
- Затем удалите этот атрибут.
- В CSS-редакторе для `td` добавьте свойство `padding: 5px;`.

- А ниже него добавьте свойство `padding-right: 30px;`.

Код в HTML

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>Отступы внутри ячеек</title>
   </head>
   <body>
      <h1>Распределение браузеров</h1>

      <table cellpadding="2">
         <tr>
            <td>Браузер</td>
            <td>Посещения</td>
            <td>Доля</td>
         </tr>
         <tr>
            <td>Mozilla 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>
```

Код в CSS

```
table {
   border-collapse: collapse;
   border: 3px solid black;
}

td {
   border: 2px solid lightgray;
}
```

Отступы между ячейками

Большинство задач по оформлению таблиц решаются с помощью работы с рамками, отступами внутри ячеек, изменения цвета фона ячеек.
Помимо внутренних отступов можно задавать отступы между ячейками таблицы.
Отступы между ячейками не работают с `border-collapse: collapse`, что достаточно логично, ведь рамки ячеек в этом режиме «склеены» и их не разорвать.
Поэтому в этом задании мы используем `border-collapse: separate`, которое «расклеивает» ячейки. На самом деле это значение по умолчанию, а мы используем его только для наглядности. Если удалить свойство `border-collapse`, то результат не изменится, ячейки будут отображаться раздельно.
Отступы между ячейками можно задать:

- с помощью атрибута `cellspacing` тега `<table>`
- или с помощью CSS-свойства `border-spacing`.

Отметим, что свойство `border-spacing` задаётся для таблицы, в отличие от `padding`, которое задаётся для ячеек.

Задание 8.

- В HTML-редакторе измените значение атрибута `cellspacing` на `5`.
- Затем удалите этот атрибут.
- В CSS-редакторе для `table` добавьте свойство `border-spacing: 5px;`.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Отступы между ячейками</title>
  </head>
  <body>
    <h1>Распределение браузеров</h1>

    <table cellspacing="3">
      <tr>
        <td>Браузер</td>
        <td>Посещения</td>
        <td>Доля</td>
      </tr>
      <tr>
        <td>Mozilla 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>
```

Код в CSS

```
table {
  border-collapse: separate;
  border: 3px solid black;
}

td {
  border: 2px solid lightgray;
  padding: 5px;
}
```

Задание

Для того, чтобы пройти это испытание, вам нужно оформить таблицу так же, как в образце. Используйте знания, полученные в пройденных заданиях.

Подсказки:

1. Цвет рамок: `lightgray`.
2. Стили для `body` менять не надо.
3. Использовались только стандартные теги.
4. Стили применять только к табличным тегам: `<table>`, `<td>`.
5. Размеры отступов кратны `5`.
6. При стилизации рамок в CSS не забывайте слово `solid`.

Если отображение в мини-браузере сильно отличается от образца, то используйте вкладку «Различия», в которой отображается то, как академия видит ваш HTML-код.

Код в HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Простая, но аккуратная таблица</title>
  </head>
  <body>
    <h1>Посещения по городам</h1>

    Город  Посещения  Страниц  Время
    СПб  199  18,02  00:13:45
    Москва  69  1,48  00:00:44
    Киев  5  13,43  00:18:07

  </body>
</html>
```

Код в CSS

```
body {
   width: 350px;
   margin: 0;
   padding: 0 10px;
   font-size: 14px;
   font-family: Arial;
}
```

Посещения по городам

Город	Посещения	Страниц	Время
СПб	199	18,02	00:13:45
Москва	69	1,48	00:00:44
Киев	5	13,43	00:18:07

Лабораторная работа 12
Разработка web-сайта с использованием Django

Для создания **веб-страниц** и **таблиц стилей** в среде **Visual Studio 2017** необходимо:

1. Запустить Visual Studio 2017.
2. В меню **Файл** выбрать команду **Создать** далее выбрать «Файл». В появившемся списке «Создать файл» выбрать «HTML-документ» или «Таблица стилей» и нажать кнопку «Создать».
3. Созданный файл следует сохранить в созданной для выполнения работы папке командой меню **Файл** «Сохранить <имя файла> как…»

Для того, чтобы открыть веб-страницу из локальной папки в браузере необходимо:

1. Открыть папку с файлами в Проводнике Windows.
2. По правой кнопке мыши вызвать для выбранного в списке файла контекстное меню и выбрать там команду «Открыть с помощью». В появившемся списке выбрать для просмотра один из установленных в системе веб-браузеров.

Для того, чтобы редактировать файлы веб-сайта:

1. В меню **Файл** выбрать команду **Открыть** далее выбрать «Файл». В появившемся окне выбрать файл и нажать кнопку «Открыть».
2. Выполнить необходимые изменения.
3. Командой меню **Файл** «Сохранить <имя файла>» сохранить изменения.

Для того, чтобы отслеживать результаты выполняемых изменений:

1. Открыть файлы для редактирования и открыть HTML-файл для просмотра в браузере.
2. Выполнять и сохранять изменения в файлах.
3. Отслеживать результаты сделанных изменений, нажимая кнопку «Обновить» в браузере (или нажимая на клавиатуре клавишу F5 или комбинацию Ctrl-R).

Для того, чтобы добавить в макеты страниц графические файлы, следует:

1. Скопировать файлы картинок в папку веб-сайта.
2. Добавить в разметку тег **img**, значением параметра **src** которого будет имя файла картинки.

Для включения ссылок в разметку страниц следует:

1. Использовать тег **a** с атрибутом **href.**

2. В качестве значение атрибута **href** для ссылок на файлы в той же папке использовать имя файла документа, на который ссылается ссылка.
3. Для ссылок на другие ресурсы в интернете в качестве значение атрибута **href** указывать полный путь, содержащий, как минимум, префикс **http://** или **https://** и доменное имя сайта, например, www.mail.ru.
   При создании разметки страниц используйте информацию по тегам HTML5 из справочника https://webref.ru/html и по стилевым таблицам из справочника https://webref.ru/css.

Задание 1

1. Создать на любом диске компьютера папку для файлов веб-сайта.
2. Создать стартовую веб-страницу **about.html** и сохранить ее в папке файлов веб-сайта. (см. теорию в подразделе.
3. Добавить в папку файл стилевой таблицы **site.css** и дополнительный HTML-файл **links.html**.
4. Подобрать текст и графические файлы по тематике будущего сайта для информационного наполнения веб-страниц. Страница **about.html** должна будет содержать общую информацию о сайте, 2-3 абзаца текста с иллюстрациями, а страница links.html – от 3-х ссылок на другие веб-ресурсы с кратким комментарием к ним.
5. Создать соответствующую правилам HTML-разметку веб-страницы **about.html**, используя не менее 10 изученных в данной теме различных тегов, предназначенных для использования в секции body HTML-документа. Страница должна содержать текст, графические изображения и ссылку на **links.html**. Страница **links.html** должна содержать ссылки на другие ресурсы.
6. Реализовать стилевое оформление веб-страниц, заполнив файл **site.css** стилевыми правилами для различных вариантов селекторов (использовать не менее 10 разичных стилевых свойств). Применить созданные стили к элементам разметки HTML-страниц, подключив стилевой файл в секции head HTML-документов. Использовать задание цвета фона, цвета и размера шрифта и другие стилевые свойства.
7. Убедиться в успешном отображении созданных страниц в браузере и в выполнении переходов по ссылкам со страниц.

1. Выберите название файла, который хранит настройки проекта в виде набора переменных
2. Базовый шаблон принято называть: (один или несколько правильных ответов)

- [ ] base.html
- [ ] layout.html
- [ ] template.html
- [ ] index.html

3. Для создания меню навигации по сайту используется класс:
4. Пакет базового приложения Django это:

- ( ) папка с тем же названием, что и проект
- ( ) app
- ( ) base

5. Файл **contact.html** веб-проекта Django в среде Visual Studio 2017 содержит тег **body**

6. Базовый шаблон включает в себя: (один или несколько правильных ответов)

- ☐ логотип
- ☐ меню
- ☐ контент
- ☐ подвал

7. Файл **layout.html** веб-проекта Django в среде Visual Studio 2017 содержит тег **body** [ ]
8. MVC определяет способ разработки программного обеспечения [ ]
9. Напишите строчными буквами название понятия, которое обозначает процесс получения HTML-страницы из шаблона, когда подставляются значения в переменные и применяются операторы [ ] нет ответа
10. Представление включает в себя только HTML-код [ ]
11. Представление включает в себя: (один или несколько правильных ответов)

- ☐ Регулярные выражения
- ☐ Переменные шаблона
- ☐ Параметры локализации
- ☐ Теги шаблона
- ☐ HTML-код

12. Какой файл определяет, что папка, в которой он расположен, является пакетом проекта: [ ]
13. Для добавления в конкретное место шаблона веб-страницы каких-либо значений используют теги шаблона [ ]
14. Шаблонные теги **{% block scripts %}…{% endblock %}** задают место вставки скриптов из представлений, которые используют конкретную Мастер-страницу [ ]
15. Значение какого параметра необходимо изменить в файле **site.css**, чтобы обеспечить отсутствие перекрытия «шапки» сайта с содержанием страниц: [ ]

Лабораторная работа 13-14

Создание web-проекта Django

Для создания **проекта программы на языке Python** в среде **Visual Studio 2017** необходимо:

1. Запустить Visual Studio 2017.
2. В меню **Файл** выбрать команду **Создать** далее выбрать «**Проект**». В появившейся панели «**Создание проекта**» выбрать в левом столбце «**Python**» и в появившемся справа списке выбрать строку «**Приложение Python**». В нижней части панели справа от поля ввода «**Расположение**» следует нажать кнопку «**Обзор**» и выбрать папку, в которой будет размещаться папка проекта. Затем закрыть панель кнопкой «**OK**».

Для того, чтобы открыть ранее созданный проект в Visual Studio 2017:

1. В меню **Файл** выбрать команду **Открыть** далее выбрать «**Решение или проект**». В появившемся окне выбора файла войти в папку проекта и выбрать файл с расширением **.sln**.
2. Если не отображается окно «Обозреватель решений» выбрать через меню **Вид** пункт «Обозреватель решений».

Для того, чтобы создать текст программы на языке Python в Visual Studio 2017:

1. Через «Обозреватель решений» открыть файл PythonApplication1.py. Исходно этот файл пуст. В окне этого файла надо набрать текст программы.
2. Сохранить текст программы командой меню **Файл** «Сохранить PythonApplication1.py».

Для того, чтобы выполнить программу:

1. Открыть проект программы в Visual Studio.
2. В меню **Проект** выполнить команду «**Запуск без отладки**».
3. В появившемся консольном окне (окне с текстовым режимом вывода) при первом его появлении рекомендуется поменять размер шрифта выводимого текста на более крупный. Для этого надо кликнуть правой кнопкой мыши по заголовку окна, в контекстном меню выбрать пункт «**Свойства**» и в появившейся панели выбрать удобный для просмотра размер шрифта в списке «**Размер**».
4. В консольном окне выполнять запрограммированные операции ввода с клавиатуры и наблюдать вывод программы на экран.

Для того, чтобы извлечь квадратный корень из числового значения:

1. Следует поместить в начало текста программы строку, подключающую пакет **math**, содержащий математические функции:
   import math
2. Для извлечения корня необходимо передать числовое значение в функцию *math.sqrt()* и использовать результат, возвращаемый этой функцией.

Для того, чтобы округлить числовое значение до второго знака после запятой:

Числовое значение следует передать функции round(), получающей два аргумента – само числовое значение и точность округления, как число знаков после запятой. Затем использовать результат, возвращаемый этой функцией.

Для того, чтобы вычислить периметр прямоугольного треугольника:

1. Следует сложить длины двух катетов прямоугольного треугольника и длину его гипотенузы.
2. Длина гипотенузы вычисляется как квадратный корень из суммы квадратов катетов.

Для того, чтобы вычислить площадь прямоугольного треугольника:

Произведение длин катетов треугольника следует поделить на 2.

Задание 1

Цель задания – получить начальные навыки программирования на языке Python.

1. Создать проект программы на языке Python в среде Visual Studio 2017.
2. Написать программу по расчету периметра и площади прямоугольного треугольника на основе ввода пользователем длин двух его катетов. Входные данные вводятся как целые числа. Программа должна выводить на экран текст, содержащий описание выводимых данных и их числовые значения. Выводимые данные должны включать в себя входные параметры и результаты вычислений. Результаты должны округляться до второго знака после запятой.
3. Выполнить программу и убедиться, что полученные в программе результаты совпадают с результатами проверочных математических вычислений.
4. В случае обнаружения ошибок, исправить ошибки и выполнить программу повторно.

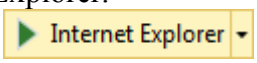
Для создания **веб-проекта Django** в среде **Visual Studio 2017** необходимо:

1. Запустить Visual Studio 2017.

2. В меню **Файл** выбрать команду **Создать** далее выбрать **Проект**. В появившемся окне «Создание проекта»:
- в левой части окна выбрать **Python** –> **Web**
- в центральной части окна выбрать **Веб-проект Django**
- внизу окна ввести имя проекта и выбрать папку для проекта
- нажать кнопку ОК

3. Установить внешние пакеты в проект: для этого выбрать пункт «Установить в виртуальном окружении». В окне «Добавление виртуального окружения» нажать кнопку «Создать».

Для того, чтобы запустить веб-приложение на выполнение в среде Visual Studio 2017, необходимо:

1. При первом запуске можно в настройке способа отладочного запуска выбрать браузер для запуска приложения – например, Internet Explorer.
2. Запустить веб-приложение, нажав кнопку 

Для того, чтобы заменить меню, заголовки и тексты стандартных макетов страниц шаблона сайта на соответствующие русскоязычные названия и тексты, необходимо:

1. В файле settings.py заменить значение параметра LANGUAGE_CODE на 'ru-ru'
2. Сохранить все веб-страницы (все html-файлы в папке \app\templates\app) в кодировке **UTF-8** одним из двух альтернативных способов:
- либо в среде Visual Studio 2017 с помощью команды **Файл > Сохранить имя.html как...** сохранить каждую веб-страницу в кодировке UTF-8 (в диалоговом окне **Сохранить файл как** необходимо выбрать из списка **Сохранить** команду **Сохранить с кодировкой** и далее выбрать в диалоговом окне **Дополнительные параметры сохранения** кодировку: **Юникод(UTF-8**, с сигнатурой) и нажать **ОК**).
- либо с помощью текстового редактора **Блокнот** все **html-страницы** сохранить в кодировке UTF-8.

3. Отредактировать код контроллера **views.py** – заменить англоязычные заголовки страниц на русскоязычные.
4. Отредактировать макеты всех страниц сайта в папке \app\templates\app. При создании разметки страниц используйте информацию по тегам HTML5 из справочника: https://webref.ru/html
Справочник по библиотеке **Bootstrap** доступен по ссылке: https://webref.ru/layout/bootstrap

Для того, чтобы поместить графический логотип сайта в верхней области страниц и, чтобы клик по логотипу приводил к переходу на главную страницу сайта, необходимо:

1. Добавить картинку логотипа сайта в папку **app\static\app\content**
2. Добавить в файле **templates\app\layout.html** в блоке после блока <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
вместо кода:
Application name
следующий код:

где вместо имя_файла_с_логотипом укажите имя файла картинки.
3. В файле стилей **app\static\app\content\site.css** для тега **body** подберите значение параметра **padding-top**, которое обеспечит отсутствие перекрытия шапки сайта с содержанием страниц.

Задание 1

Цель задания – освоить технологию создания веб-проекта Django в среде Visual Studio 2017.

1. Создать веб-проект Django

2. Запустить веб-приложение на выполнение. Завершить работу веб-приложения.
3. Изучить структуру веб-проекта в «Обозревателе решений» .
4. Заменить меню, заголовки и тексты стандартных макетов страниц шаблона веб-сайта на соответствующие русскоязычные названия и тексты в соответствии с выбранной темой (см. теорию в подразделе .
5. Найти или создать самостоятельно графический файл (с расширением jpg или png) для использования в качестве логотипа Вашего веб-сайта. Поместить графический логотип сайта в верхней области страниц. Клик по логотипу должен приводить к переходу на главную страницу сайта. (см. теорию в подразделе.
6. Проверить правильность отображения контента веб-страниц в браузере и возможность перехода на главную страницу при нажатии на логотип.