МИНИСТЕРСТВО НАУКИ, ВЫСШЕГО ОБРАЗОВАНИЯ И ИННОВАЦИЙ КЫРГЫЗСКОЙ РЕСПУБЛИКИ
КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ им. И.АРАБАЕВА
ИНСТИТУТ НОВЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
ОТДЕЛЕНИЕ СРЕДНЕГО-ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«УТВЕРЖДАЮ»
Директор ИНИТ
КГУ им. И. Арабаева
к.т.н., и. о. доц. Керимов У.Т.
«___» _____ 2025г.

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС

по дисциплине: **«Основы веб технологии»**
для студентов специальности: 230109 «Программное обеспечение вычислительной техники и автоматизированных систем», 230701 «Прикладная информатика (по отраслям)», 220206 «Автоматизированные системы обработки информации и управления (по отраслям)»
форма обучения: очное
Учебно-методический комплекс составлен на основе Государственного Образовательного Стандарта среднего профессионального образования КР
Учебно-методический комплекс разработал: магистр-преподаватель отделения СПО ИНИТ КГУ имени И. Арабаева Анарбеков Табылды Райымбекович
Соавторы: Токтомуш у Баян

Бишкек-2025

МИНИСТЕРСТВО НАУКИ, ВЫСШЕГО ОБРАЗОВАНИЯ И ИННОВАЦИЙ КЫРГЫЗСКОЙ РЕСПУБЛИКИ
КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ им. И.АРАБАЕВА
ИНСТИТУТ НОВЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
ОТДЕЛЕНИЕ СРЕДНЕГО-ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«УТВЕРЖДАЮ»
Директор ИНИТ
КГУ им. И. Арабаева
к.т.н., и. о. доц. Керимов У.Т.
«01» 09 2025г.

РАБОЧАЯ ПРОГРАММА

по дисциплине: **«Основы веб технологии»**
для студентов специальности: 230109 «Программное обеспечение вычислительной техники и автоматизированных систем», 230701 «Прикладная информатика (по отраслям)», 220206 «Автоматизированные системы обработки информации и управления (по отраслям)».

форма обучения: очное/заочное
институт: ИНИТ
отделение: ОСПО ИНИТ
курс: 2/3
семестр: 3/5/6
аттестация (семестр): 5
экзамен (семестр): 3/5/6
всего часов по учебному плану: 72
из них:
-лекции:44
-практические: 28
-самостоятельная работа: 24

Рабочая программа составлена в соответствии с требованиями Государственного Образовательного Стандарта среднего профессионального образования КР

Рабочую программу разработал: магистр-преподаватель отделения СПО ИНИТ КГУ имени И. Арабаева Анарбеков Табылды Райымбекович
Соавторы: Токтомуш у Баян

Рассмотрена и утверждена на заседании
ОСПО ИНИТ КГУ им. И. Арабаева
Протокол № 1
от «02» 09 2025г.

Зав. отделением: Н.С. Сейтказиева

Одобрено учебно-методическим советом
ИНИТ КГУ им. И. Арабаева
Протокол № 1
от «04» 09 2025г.

Председатель УМС ИНИТ:

2

Содержание

Оглавление

ЦЕЛИ И ЗАДАЧИ УЧЕБНОЙ ДИСЦИПЛИНЫ

Целью учебной дисциплины «Основы Web-технологии» является освоение инструментальных средств и информационных технологий, позволяющих создавать Web-страницы и Web-сайты**.**

Основными **задачами** данного учебного курса являются:

- уяснение основ новых информационных технологий применительно к созданию Web-сайтов;
- знание современного состояния и направления развития прикладных программных средств по изучаемому курсу;
- умение выбирать необходимые технологические средства на множестве информационных технологий при решении проблемы;
- приобретение практических навыков работы с пакетами прикладных программ, позволяющих автоматизировать некоторые приемы при создании Web-сайтов**.**

Изучение дисциплины «Основы Web-технологии» предусматривает включение следующих **вопросов**:

– обзор языков разметки;
– обзор Web-редакторов;
– ознакомление с Web-редакторами.
– изучение структуры HTML-кода;
– обзор Web-браузеров.

Требования к уровню освоения дисциплины.

В результате изучения предмета «Основы Web-технологии» студент должен **знать**:

1. основы новых информационных технологий применительно к созданию **Web-сайтов**;
2. современное состояние и направление развития прикладных программных средств по дисциплине;

Студент, прошедший курс обучения, должен **уметь**:

1. выбирать необходимые технологические средства;
2. создавать Web-страницы
3. понимать HTML-код Web-страницы;
4. создавать простые иллюстрации к Web-страницам;
5. создавать фотоальбом.

ТЕМАТИЧЕСКИЙ ПЛАН КУРСА «ОСНОВЫ WEB-ТЕХНОЛОГИИ»

Наименование разделов, дисциплин и тем				
	Лекции	**Практические занятия**	**СРС**	**Всего часов**
Очная форма обучения				
Лекция 1: Введение в HTML. Структура документа	2			
Лекция 2: Теги заголовков, параграфы и текстовые теги	2			
Лекция 3: Списки в HTML	2			
Лекция 4: Работа со ссылками и изображениями	2			
Лекция 5: Семантические теги	2			
Лекция 6: Таблицы в HTML	2			
Лекция 7: Пользовательские формы	2			
Лекция 8: Добавление графики, аудио и видео	2			
Лекция 9: Введение в CSS и способы подключения	2			
Лекция 10: Свойства CSS	2			
Лекция 11: Блочная модель CSS	2			
Лекция 12: Работа с блоками	2			
Лекция 13: Цветовая модель и работа с фоном	2			
Лекция 14: Создание страницы с использованием стилей	2			
Лекция 15: Flexbox: свойства и примеры использования	2			
Лекция 16: Grid Layout: создание сеток с помощью CSS Grid	2			
Лекция 17: Псевдоклассы и псевдоэлементы	2			
Лекция 18: Анимации и переходы	2			
Лекция 19: Адаптивная и отзывчивая вёрстка	2			

Лекция 20: Flexbox в адаптивной вёрстке	2			
Лекция 21: Основы SEO для HTML	2			
Лекция 22: Публикация сайта (GitHub Pages, Netlify)	2			
Практическая работа №1. Основы HTML		2		
Практическая работа №2. Работа со ссылками и изображениями		2		
Практическая работа №3. Семантические теги и структура страницы		2		
Практическая работа №4. Таблицы в HTML		2		
Практическая работа №5. Создание и стилизация пользовательских форм		2		
Практическая работа №6. Подключение CSS и работа с основными свойствами		2		
Практическая работа №7. Блочная модель и позиционирование элементов		2		
Практическая работа №8 Flexbox на практике		2		
Практическая работа №9 Grid Layout на практике		2		
Практическая работа №10 Анимации и переходы		2		
Практическая работа №11 Адаптивная вёрстка с медиа-запросами		2		
Практическая работа №12 Итоговый проект		6		
ИТОГО по курсу:	44	28		
Форма контроля	экзамен			
Всего часов				

Распределение баллов по модулям и видам учебных занятий

№	Этапы проверки	Вид средства проверки	Баллы
1	1 модуль	Проверка практических заданий. Устный, тестирование. Посещение занятий.	100
2	2 модуль	Проверка практических заданий. Тестирование. Посещение занятий.	100
3	Итоговый контроль: • Практическое занятие; • СРС.	Контрольные и графические работы, рефераты, презентации, СРС, практические задания. Тестирование. Посещение занятий.	100
	Итого средний балл		**100**

Итоговое распределение баллов по модулям

		Удовлетворительно	Хорошо	Отлично
Модуль 1 – 100 б.		60-79	80-89	90-100
Модуль 2 – 100 б.		60-79	80-89	90-100
Практическое занятие – 50 б.	Итоговый контроль	60-79	80-89	90-100
СРС – 50 б.				

Тема 1: Введение в HTML. Структура документа

Цель лекции:
Познакомить студентов с основами HTML, показать структуру веб-страницы и роль основных тегов: <!DOCTYPE>, <html>, <head>, <title>, <body>.

Теги и атрибуты:

- <!DOCTYPE> — определяет версию HTML документа.
  Пример:

```
<!DOCTYPE html>
```

- <html> — корневой тег документа.
  Атрибут:
- lang — указывает язык документа
  Пример:

```
<html lang="ru">
  ...
</html>
```

- <head> — секция для метаданных страницы.
  Пример:

```
<head>
```

```
  ...
</head>
```

- <title> — название страницы, отображается во вкладке браузера.
  Пример:

```
<title>Моя первая страница</title>
```

- <meta> — метаданные документа.
  Атрибут:
- charset — задаёт кодировку символов
  Пример:

```
<meta charset="UTF-8">
```

- <body> — основное содержимое страницы.
  Атрибуты:
- class — имя класса для CSS-стилей
- id — идентификатор элемента
  Пример:

```
<body id="main-body" class="page">
  ...
</body>
```

- <h1> — заголовок первого уровня.
  Атрибуты:
- align — выравнивание текста (left, center, right)
- class — имя класса для CSS
  Пример:

```
<h1 align="center" class="title">Привет!</h1>
```

- <p> — абзац текста.
  Атрибуты:
- align — выравнивание текста
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<p align="left" id="main-text" class="intro">Это мой первый HTML-документ.</p>
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Моя первая страница</title>
  </head>
  <body id="main-body" class="page">
    <h1 align="center" class="title">Привет!</h1>
```

```
   <p align="left" id="main-text" class="intro">Это мой первый HTML-документ.</p>
  </body>
</html>
```

Классная работа:

- Создать мини-страницу «О себе».
- Добавить заголовок <h1> с вашим именем.
- Добавить абзац <p> с краткой информацией о себе.
- Использовать атрибуты id и class для элементов <body> и <p>.

Домашнее задание:

- Создать страницу «О себе» с заголовком <h1> и абзацем <p>.
- Добавить <title> для страницы.
- Сдать HTML-файл или скриншот страницы.

Тема 2: Теги заголовков, параграфы и текстовые теги

Цель лекции:
Научить студентов использовать заголовки (<h1>–<h6>), абзацы (<p>) и текстовые теги для выделения информации: , , <b>, <i>.

Теги и атрибуты:

- <h1> — заголовок первого уровня.
  Атрибуты:
- align — выравнивание текста (left, center, right)
- class — имя класса для CSS
  Пример:

```
<h1 align="center" class="title">Главный заголовок</h1>
```

- <h2> — заголовок второго уровня.
  Пример:

```
<h2 class="subtitle">Подзаголовок</h2>
```

- <h3>–<h6> — заголовки третьего-шестого уровня.
  Пример:

```
<h3> Заголовок уровня 3 </h3>
<h4>Заголовок уровня 4</h4>
<h5>Заголовок уровня 5</h5>
<h6>Заголовок уровня 6</h6>
```

- <p> — абзац текста.
  Атрибуты:
- align — выравнивание текста

- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<p align="left" class="intro" id="main-text">Это пример абзаца текста.</p>
```

- — выделение текста как важного (жирный по смыслу).
  Пример:

```
<p>Это <strong>важный</strong> текст.</p>
```

- — выделение текста как эмоционально значимого (курсив по смыслу).
  Пример:

```
<p>Это <em>эмоционально выделенный</em> текст.</p>
```

- <b> — выделение текста жирным шрифтом (без смысловой нагрузки).
  Пример:

```
<p>Это <b>жирный</b> текст.</p>
```

- <i> — выделение текста курсивом (без смысловой нагрузки).
  Пример:

```
<p>Это <i>курсив</i> текст.</p>
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
 <head>
  <meta charset="UTF-8">
  <title>Заголовки и текстовые теги</title>
 </head>
 <body>
  <h1 align="center" class="title">Главный заголовок</h1>
  <h2 class="subtitle">Подзаголовок</h2>
  <h3>Заголовок уровня 3</h3>
  <p align="left" class="intro" id="main-text">
   Это <strong>важный</strong> текст, <em>эмоционально
выделенный</em>,
   а это <b>жирный</b> и <i>курсивный</i> текст.
  </p>
 </body>
</html>
```

Классная работа:

- Создать страницу с заголовками <h1>–<h3>.
- Добавить несколько абзацев <p> с использованием тегов , , <b>, <i>.
- Попробовать использовать атрибуты align, class и id для элементов.

Домашнее задание:

- Создать страницу с заголовками разного уровня и абзацами, используя текстовые теги для выделения важных частей текста.
- Сдать HTML-файл или скриншот страницы.

Тема 3: Списки в HTML

Цель лекции:
Научить студентов создавать маркированные и нумерованные списки с помощью тегов <ul>, <ol> и <li>, а также применять атрибуты для стилизации списка.

Теги и атрибуты:

- — маркированный (ненумерованный) список.
  Атрибуты:
- type — тип маркера (circle, disc, square)
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<ul type="circle" class="my-list" id="list1">
  <li>Пункт 1</li>
  <li>Пункт 2</li>
  <li>Пункт 3</li>
</ul>
```

- — нумерованный список.
  Атрибуты:
- type — тип нумерации (1, A, a, I, i)
- start — с какого числа начинать нумерацию
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<ol type="1" start="3" class="ordered-list" id="list2">
  <li>Третий пункт</li>
  <li>Четвёртый пункт</li>
  <li>Пятый пункт</li>
</ol>
```

- — элемент списка (внутри <ul> или <ol>).
  Атрибуты:
- class — имя класса для CSS

- id — идентификатор элемента
  Пример:

```
<li class="item" id="item1">Первый элемент</li>
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
 <head>
  <meta charset="UTF-8">
  <title>Списки в HTML</title>
 </head>
 <body>
  <h1>Примеры списков</h1>

  <h2>Маркированный список</h2>
  <ul type="square" class="my-list" id="list1">
   <li>Пункт 1</li>
   <li>Пункт 2</li>
   <li>Пункт 3</li>
  </ul>

  <h2>Нумерованный список</h2>
  <ol type="A" start="3" class="ordered-list" id="list2">
   <li>Третий пункт</li>
   <li>Четвёртый пункт</li>
   <li>Пятый пункт</li>
  </ol>
 </body>
</html>
```

Классная работа:

- Создать страницу с **маркированным** и **нумерованным** списком.
- Использовать атрибуты type, class и id для списков и элементов.
- Попробовать разные типы маркеров и нумерации.

Домашнее задание:

- Сделать страницу с одним маркированным и одним нумерованным списком.
- Добавить хотя бы три элемента <li> в каждый список.
- Сдать HTML-файл или скриншот страницы.

Тема 4: Работа со ссылками и изображениями

Цель лекции:
Научить студентов создавать гиперссылки с помощью тега <a> и добавлять изображения с помощью тега <img>. Рассмотреть основные атрибуты этих тегов.

Теги и атрибуты:.

- <a> — тег для создания гиперссылки.
  Атрибуты:
- href — адрес ссылки
- target — где открывать ссылку (_blank — новое окно/вкладка)
- title — всплывающая подсказка при наведении
  Пример:

```
<a href="https://example.com" target="_blank" title="Перейти на Example">Ссылка на Example</a>
```

- <img> — тег для вставки изображения.
  Атрибуты:
- src — путь к изображению
- alt — альтернативный текст для изображения
- width — ширина изображения
- height — высота изображения
- title — всплывающая подсказка
  Пример:

```
<img src="image.jpg" alt="Описание изображения" width="300" height="200" title="Пример изображения">
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
 <head>
  <meta charset="UTF-8">
  <title>Ссылки и изображения</title>
 </head>
 <body>
  <h1>Примеры ссылок и изображений</h1>

  <h2>Ссылка</h2>
  <p>
   Посетите сайт:
   <a href="https://example.com" target="_blank" title="Перейти на Example">Example.com</a>
  </p>
```

```
  <h2>Изображение</h2>
  <p>
    Пример изображения:
    <img src="image.jpg" alt="Описание изображения" width="300"
height="200" title="Пример изображения">
  </p>
 </body>
</html>
```

Классная работа:

- Создать страницу с гиперссылкой <a> на любой сайт.
- Добавить изображение <img> с атрибутами src, alt, width, height, title.
- Попробовать открыть ссылку в новом окне с помощью target="_blank".

Домашнее задание:

- Сделать страницу с одной ссылкой на интересный сайт и одним изображением.

  Сдать HTML-файл или скриншот страницы.

Тема 5: Семантические теги

Цель лекции:
Познакомить студентов с семантическими тегами HTML и показать, как структурировать страницу для удобства чтения, SEO и доступности.

Теги и атрибуты:

- <header> — верхняя часть страницы или раздела, обычно содержит заголовок или меню.
  Атрибуты:
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<header class="site-header" id="main-header">
 <h1>Мой сайт</h1>
</header>
```

- <footer> — нижняя часть страницы или раздела, обычно содержит контакты или авторские права.
  Атрибуты:
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<footer class="site-footer" id="main-footer">
  <p>© 2025 Мой сайт</p>
</footer>
```

- <nav> — блок навигации по сайту, содержит ссылки на страницы.
  Атрибуты:
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<nav class="main-nav" id="nav">
  <a href="index.html">Главная</a>
  <a href="about.html">О сайте</a>
</nav>
```

- <section> — раздел страницы, может содержать заголовки и контент.
  Атрибуты:
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<section class="intro" id="section1">
  <h2>Введение</h2>
  <p>Это секция с вводной информацией.</p>
</section>
```

- <article> — самостоятельный блок контента, например, статья или новость.
  Атрибуты:
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<article class="news-item" id="article1">
  <h2>Новость дня</h2>
  <p>Текст новости...</p>
</article>
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Семантические теги</title>
  </head>
  <body>
    <header class="site-header" id="main-header">
      <h1>Мой сайт</h1>
    </header>
```

```
    <nav class="main-nav" id="nav">
      <a href="index.html">Главная</a>
      <a href="about.html">О сайте</a>
    </nav>

    <section class="intro" id="section1">
      <h2>Введение</h2>
      <p>Это секция с вводной информацией.</p>
    </section>

    <article class="news-item" id="article1">
      <h2>Новость дня</h2>
      <p>Текст новости...</p>
    </article>

    <footer class="site-footer" id="main-footer">
      <p>© 2025 Мой сайт</p>
    </footer>
  </body>
</html>
```

Классная работа:

- Создать страницу с семантической структурой: <header>, <nav>, <section>, <article>, <footer>.
- Добавить заголовки и текстовый контент в каждый блок.
- Использовать атрибуты class и id для каждого элемента.

Домашнее задание:

- Сделать страницу с семантической структурой и заполнить контентом хотя бы один <section> и один <article>.
- Сдать HTML-файл или скриншот страницы.

Тема 6: Таблицы в HTML

Цель лекции:
Научить студентов создавать таблицы, заполнять их данными и использовать основные атрибуты для форматирования.

Теги и атрибуты:

- <table> — создаёт таблицу.
  Атрибуты:
- border — ширина рамки таблицы

- cellpadding — внутренние отступы ячеек
- cellspacing — расстояние между ячейками
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<table border="1" cellpadding="5" cellspacing="2" class="my-table" id="table1">
  ...
</table>
```

- <tr> — строка таблицы.
  Атрибуты:
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<tr class="row" id="row1">
  ...
</tr>
```

- <td> — ячейка таблицы (данные).
  Атрибуты:
- colspan — объединение ячеек по горизонтали
- rowspan — объединение ячеек по вертикали
- align — выравнивание текста (left, center, right)
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<td colspan="2" align="center" class="cell" id="cell1">Ячейка 1</td>
```

- <th> — заголовочная ячейка таблицы (жирный текст по умолчанию, центрированный).
  Атрибуты:
- colspan — объединение ячеек по горизонтали
- rowspan — объединение ячеек по вертикали
- align — выравнивание текста
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<th colspan="2" align="center" class="header-cell" id="th1">Заголовок</th>
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
```

```
    <meta charset="UTF-8">
    <title>Таблицы в HTML</title>
  </head>
  <body>
    <h1>Пример таблицы</h1>
    <table border="1" cellpadding="5" cellspacing="2" class="my-table"
id="table1">
      <tr>
        <th colspan="2" align="center" class="header-cell" id="th1">Заголовок
таблицы</th>
      </tr>
      <tr class="row" id="row1">
        <td class="cell" id="cell1">Ячейка 1</td>
        <td class="cell" id="cell2">Ячейка 2</td>
      </tr>
      <tr class="row" id="row2">
        <td class="cell" id="cell3">Ячейка 3</td>
        <td class="cell" id="cell4">Ячейка 4</td>
      </tr>
    </table>
  </body>
</html>
```

Классная работа:

- Создать таблицу с заголовком <th> и как минимум двумя строками <tr> и ячейками <td>.
- Использовать атрибуты border, cellpadding, cellspacing, colspan и rowspan.
- Добавить class и id для элементов таблицы.

Домашнее задание:

- Сделать страницу с таблицей, заполненной данными (например, расписание или список товаров).
- Сдать HTML-файл или скриншот страницы.

Тема 7: Пользовательские формы

Цель лекции:
Научить студентов создавать формы для ввода данных и использовать основные элементы: текстовые поля, кнопки, выпадающие списки и текстовые области.

Теги и атрибуты:

- <form> — создаёт форму для отправки данных.
  Атрибуты:
- action — адрес, куда отправляются данные
- method — метод отправки данных (get, post)
- enctype — кодировка отправки данных
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<form action="/submit" method="post" enctype="application/x-www-form-urlencoded" class="my-form" id="form1">
  ...
</form>
```

- <input> — поле ввода данных.
  Атрибуты:
- type — тип поля (text, password, email, checkbox, radio, submit и др.)
- name — имя поля для отправки данных
- value — значение по умолчанию
- placeholder — подсказка внутри поля
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<input type="text" name="username" value="" placeholder="Введите имя" class="input-field" id="input1">
```

- <textarea> — многострочное текстовое поле.
  Атрибуты:
- name — имя поля для отправки данных
- rows — количество строк
- cols — ширина поля
- placeholder — подсказка
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<textarea name="message" rows="4" cols="30" placeholder="Введите сообщение" class="textarea" id="message1"></textarea>
```

- <select> — выпадающий список.
  Атрибуты:
- name — имя поля
- class — имя класса для CSS

- id — идентификатор элемента
  Пример:

```
<select name="options" class="dropdown" id="select1">
  <option value="1">Опция 1</option>
  <option value="2">Опция 2</option>
  <option value="3">Опция 3</option>
</select>
```

- <button> — кнопка.
  Атрибуты:
- type — тип кнопки (submit, reset, button)
- disabled — блокировка кнопки
- name — имя для отправки данных
- value — значение кнопки
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<button type="submit" class="btn" id="submitBtn">Отправить</button>
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Форма HTML</title>
  </head>
  <body>
    <h1>Пример формы</h1>
    <form action="/submit" method="post" class="my-form" id="form1">
      <label for="input1">Имя:</label>
      <input type="text" name="username" placeholder="Введите имя"
class="input-field" id="input1"><br><br>

      <label for="message1">Сообщение:</label>
      <textarea name="message" rows="4" cols="30" placeholder="Введите
сообщение" class="textarea" id="message1"></textarea><br><br>

      <label for="select1">Выберите опцию:</label>
      <select name="options" class="dropdown" id="select1">
        <option value="1">Опция 1</option>
        <option value="2">Опция 2</option>
        <option value="3">Опция 3</option>
      </select><br><br>

      <button type="submit" class="btn" id="submitBtn">Отправить</button>
```

```
  </form>
 </body>
</html>
```

Классная работа:

- Создать форму с полями <input>, <textarea> и <select>.
- Добавить кнопку <button> для отправки формы.
- Использовать атрибуты name, id, class, placeholder и другие по необходимости.

Домашнее задание:

- Сделать страницу с формой для обратной связи: имя, email, сообщение и выбор темы.
- Сдать HTML-файл или скриншот страницы.

Тема 8: Добавление графики, аудио и видео

Цель лекции:
Научить студентов добавлять мультимедийный контент на веб-страницу с помощью HTML: изображения, аудио и видео.

Теги и атрибуты:

- <img> — вставка изображения.
  Атрибуты:
- src — путь к изображению
- alt — альтернативный текст
- width — ширина изображения
- height — высота изображения
- title — всплывающая подсказка
  Пример:

```
<img src="image.jpg" alt="Описание изображения" width="300" height="200" title="Пример изображения">
```

- <audio> — вставка аудио-файла.
  Атрибуты:
- src — путь к аудио
- controls — отображение панели управления
- autoplay — автоматическое воспроизведение
- loop — зацикливание
- class — имя класса для CSS

- id — идентификатор элемента
  Пример:

```
<audio src="audio.mp3" controls autoplay loop class="audio-player" id="audio1"></audio>
```

- <video> — вставка видео-файла.
  Атрибуты:
- src — путь к видео
- controls — отображение панели управления
- autoplay — автоматическое воспроизведение
- loop — зацикливание
- width — ширина видео
- height — высота видео
- class — имя класса для CSS
- id — идентификатор элемента
  Пример:

```
<video src="video.mp4" controls width="640" height="360" class="video-player" id="video1"></video>
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
 <head>
  <meta charset="UTF-8">
  <title>Мультимедиа в HTML</title>
 </head>
 <body>
  <h1>Примеры мультимедиа</h1>

  <h2>Изображение</h2>
  <img src="image.jpg" alt="Описание изображения" width="300" height="200" title="Пример изображения">

  <h2>Аудио</h2>
  <audio src="audio.mp3" controls class="audio-player" id="audio1"></audio>

  <h2>Видео</h2>
  <video src="video.mp4" controls width="640" height="360" class="video-player" id="video1"></video>
 </body>
</html>
```

Классная работа:

- Добавить на страницу одно изображение <img>.

- Вставить аудио <audio> с панелью управления.
- Вставить видео <video> с панелью управления.
- Использовать атрибуты src, width, height, controls и id/class.

Домашнее задание:

- Сделать страницу с мультимедийным контентом: картинка, аудио и видео.
- Сдать HTML-файл или скриншот страницы.

Тема 9: Введение в CSS и способы подключения

Цель лекции:
Научить студентов использовать CSS для стилизации веб-страниц и рассмотреть три способа подключения стилей: встроенные (inline), внутренние (internal) и внешние (external).

Способы подключения CSS:

- **Inline (встроенные стили)** — стили задаются прямо в HTML-элементе с помощью атрибута style.
  Пример:

```
<p style="color: red; font-size: 16px;">Это текст с красным цветом и размером 16px.</p>
```

- **Internal (внутренние стили)** — стили прописываются внутри тега <style> в <head> документа.
  Пример:

```
<head>
  <style>
    p {
      color: blue;
      font-size: 18px;
    }
    h1 {
      text-align: center;
      color: green;
    }
  </style>
</head>
<body>
  <h1>Заголовок</h1>
  <p>Текст с внутренним стилем.</p>
</body>
```

- **External (внешние стили)** — стили находятся в отдельном файле CSS, подключаются с помощью <link> в <head>.
  Пример:
- **Файл style.css:**

```
body {
  background-color: #f0f0f0;
}
h1 {
  color: navy;
}
p {
  color: maroon;
  font-size: 16px;
}
```

- **HTML-файл:**

```
<head>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>Заголовок</h1>
  <p>Текст с внешним стилем.</p>
</body>
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Подключение CSS</title>
    <style>
      h1 {
        text-align: center;
        color: green;
      }
      p.internal {
        color: blue;
        font-size: 18px;
      }
    </style>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <h1>Пример CSS</h1>

    <p style="color: red; font-size: 16px;">Текст с inline-стилем.</p>
    <p class="internal">Текст с внутренним стилем.</p>
```

```
    <p class="external">Текст с внешним стилем.</p>
  </body>
</html>
```

Классная работа:

- Создать страницу с заголовком <h1> и тремя абзацами <p>.
- Использовать **встроенный**, **внутренний** и **внешний** CSS для стилизации текста.
- Попробовать менять цвет текста, размер шрифта и выравнивание.

Домашнее задание:

- Сделать страницу с заголовком и абзацами, применяя все три способа подключения CSS.
- Сдать HTML-файл и CSS-файл или скриншот страницы.

Тема 10: Свойства CSS

Цель лекции:
Научить студентов использовать основные свойства CSS для стилизации текста, изменения размеров, отступов и добавления границ элементов.

Основные свойства CSS:

- **Цвет текста и фона**
- color — цвет текста
- background-color — цвет фона элемента
  Пример:

```
p {
  color: red;
  background-color: yellow;
}
```

- **Шрифты и текст**
- font-family — семейство шрифта
- font-size — размер шрифта
- font-weight — толщина шрифта (normal, bold)
- text-align — выравнивание текста (left, center, right)
  Пример:

```
h1 {
  font-family: Arial, sans-serif;
  font-size: 24px;
  font-weight: bold;
  text-align: center;
```

```
}
```

- **Размеры элементов**
- width — ширина
- height — высота
  Пример:

```
div {
  width: 300px;
  height: 150px;
  background-color: lightblue;
}
```

- **Отступы и поля**
- margin — внешние отступы
- padding — внутренние отступы
  Пример:

```
div {
  margin: 20px;
  padding: 10px;
}
```

- **Границы элементов**
- border — общие границы
- border-width — ширина границы
- border-style — стиль границы (solid, dashed, dotted)
- border-color — цвет границы
  Пример:

```
div {
  border: 2px solid black;
}
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Свойства CSS</title>
    <style>
      h1 {
        font-family: Arial, sans-serif;
        font-size: 24px;
        font-weight: bold;
        text-align: center;
        color: navy;
      }
      p {
```

```
      color: red;
      background-color: yellow;
      padding: 10px;
      margin: 15px;
      border: 2px dashed green;
    }
    div.box {
      width: 300px;
      height: 150px;
      background-color: lightblue;
      margin: 20px auto;
      padding: 10px;
      border: 2px solid black;
    }
  </style>
 </head>
 <body>
  <h1>Пример CSS-свойств</h1>

  <p>Абзац с цветом текста, фоном, отступами и границей.</p>

  <div class="box">
    Блок с размерами, внутренними и внешними отступами, границей и фоном.
  </div>
 </body>
</html>
```

Классная работа:

- Создать страницу с заголовком <h1>, абзацем <p> и блоком .
- Использовать свойства color, background-color, font-family, font-size, font-weight, text-align, width, height, margin, padding, border.

Домашнее задание:

- Сделать страницу с текстом и блоком, применяя все рассмотренные свойства CSS.
- Сдать HTML-файл или скриншот страницы.

Тема 11: Блочная модель CSS

Цель лекции:
Научить студентов понимать и использовать блочную модель CSS для управления размерами, отступами и границами элементов на странице.

Основные свойства блочной модели:

- **width и height** — ширина и высота блока.
  Пример:

```
div {
  width: 300px;
  height: 150px;
  background-color: lightblue;
}
```

- **padding** — внутренние отступы (пространство между содержимым и границей).
  Пример:

```
div {
  padding: 20px;
}
```

- **border** — граница блока.
- border-width — толщина
- border-style — стиль (solid, dashed, dotted)
- border-color — цвет
  Пример:

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

- **margin** — внешние отступы (пространство между блоком и другими элементами).
  Пример:

```
div {
  margin: 30px;
}
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Блочная модель CSS</title>
    <style>
      div.box {
        width: 300px;
        height: 150px;
        padding: 20px;
        border: 3px solid black;
        margin: 30px auto;
        background-color: lightgreen;
      }
      p {
```

```
      margin: 10px;
      padding: 5px;
      border: 2px dashed blue;
      background-color: lightyellow;
    }
  </style>
 </head>
 <body>
  <h1>Пример блочной модели</h1>

  <div class="box">
   <p>Внутри блока есть отступы, граница и внешний отступ.</p>
  </div>
 </body>
</html>
```

Классная работа:

- Создать блок с текстом <p> внутри.
- Применить свойства: width, height, padding, border, margin.
- Попробовать изменять значения и наблюдать изменения внешнего вида.

Домашнее задание:

- Сделать страницу с одним блоком и текстом внутри, используя все свойства блочной модели.
- Сдать HTML-файл или скриншот страницы.

Тема 12: Работа с блоками

Цель лекции:
Научить студентов управлять расположением и отображением блоков на странице с помощью CSS-свойств display, position, float и flexbox.

Основные свойства:

- **display** — определяет, как элемент отображается на странице.
- block — блочный элемент
- inline — встроенный элемент
- inline-block — комбинированный режим
- none — скрыть элемент
  Пример:

```
div.block {
 display: block;
```

```
  width: 200px;
  height: 100px;
  background-color: lightblue;
}

span.inline {
  display: inline;
  color: red;
}

div.hidden {
  display: none;
}
```

- **position** — управляет позиционированием элемента.
- static — стандартное положение (по умолчанию)
- relative — относительно своего положения
- absolute — относительно ближайшего позиционированного родителя
- fixed — фиксированное положение на экране
  Пример:

```
div.relative {
  position: relative;
  top: 20px;
  left: 30px;
  background-color: lightgreen;
  width: 150px;
  height: 80px;
}
```

- **float** — обтекание элементов.
- left — слева
- right — справа
  Пример:

```
img {
  float: left;
  margin-right: 10px;
}
```

- **Flexbox** — гибкая верстка блоков.
- display: flex — активирует flex-контейнер
- justify-content — выравнивание по горизонтали (flex-start, center, flex-end, space-between)
- align-items — выравнивание по вертикали (flex-start, center, flex-end)
  Пример:

```
div.flex-container {
  display: flex;
```

```
  justify-content: space-between;
  align-items: center;
  background-color: lightgray;
  padding: 10px;
}

div.flex-item {
  background-color: lightcoral;
  width: 100px;
  height: 50px;
  text-align: center;
  line-height: 50px;
}
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Работа с блоками</title>
    <style>
      div.block {
        display: block;
        width: 200px;
        height: 100px;
        background-color: lightblue;
        margin-bottom: 10px;
      }

      div.relative {
        position: relative;
        top: 20px;
        left: 30px;
        background-color: lightgreen;
        width: 150px;
        height: 80px;
      }

      img {
        float: left;
        margin-right: 10px;
      }

      div.flex-container {
        display: flex;
        justify-content: space-between;
        align-items: center;
```

```
      background-color: lightgray;
      padding: 10px;
      margin-top: 20px;
    }

    div.flex-item {
      background-color: lightcoral;
      width: 100px;
      height: 50px;
      text-align: center;
      line-height: 50px;
    }
  </style>
 </head>
 <body>
  <h1>Примеры работы с блоками</h1>

  <div class="block">Блочный элемент</div>

  <div class="relative">Относительное позиционирование</div>

  <img src="image.jpg" alt="Картинка" width="100" height="100">
  <p>Текст обтекает изображение слева.</p>

  <div class="flex-container">
   <div class="flex-item">Элемент 1</div>
   <div class="flex-item">Элемент 2</div>
   <div class="flex-item">Элемент 3</div>
  </div>
 </body>
</html>
```

Классная работа:

- Создать страницу с блочными элементами, изображением с обтеканием и flex-контейнером с тремя элементами.
- Применить свойства display, position, float и flexbox.

Домашнее задание:

- Сделать страницу с блочной структурой: блоки, изображение с обтеканием, flex-контейнер с элементами.
- Сдать HTML-файл или скриншот страницы.

Тема 13: Цветовая модель и работа с фоном

Цель лекции:
Научить студентов использовать цвета, градиенты и скругленные углы для стилизации элементов на веб-странице.

Основные свойства CSS:

- **Цвет текста и фона**
- color — цвет текста
- background-color — цвет фона элемента
  Пример:

```
p {
  color: white;
  background-color: darkblue;
}
```

- **Градиенты**
- background: linear-gradient(...) — линейный градиент
- background: radial-gradient(...) — радиальный градиент
  Пример линейного градиента:

```
div {
  width: 300px;
  height: 150px;
  background: linear-gradient(to right, red, yellow);
}
```

Пример радиального градиента:

```
div {
  width: 300px;
  height: 150px;
  background: radial-gradient(circle, blue, green);
}
```

- **Скругление углов**
- border-radius — задаёт скругление углов элемента
  Пример:

```
div {
  width: 200px;
  height: 100px;
  background-color: lightcoral;
  border-radius: 20px;
}
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
```

```
<head>
  <meta charset="UTF-8">
  <title>Фон и градиенты</title>
  <style>
    p {
      color: white;
      background-color: darkblue;
      padding: 10px;
      border-radius: 5px;
    }

    div.linear-gradient {
      width: 300px;
      height: 150px;
      background: linear-gradient(to right, red, yellow);
      margin-bottom: 20px;
      border-radius: 10px;
    }

    div.radial-gradient {
      width: 300px;
      height: 150px;
      background: radial-gradient(circle, blue, green);
      border-radius: 50%;
    }
  </style>
</head>
<body>
  <h1>Примеры работы с фоном</h1>

  <p>Текст с фоном и скруглением углов</p>

  <div class="linear-gradient"></div>

  <div class="radial-gradient"></div>
</body>
</html>
```

Классная работа:

- Создать страницу с абзацем <p> с цветом текста и фоном.
- Создать блок с линейным градиентом и радиальный градиент.
- Применить свойство border-radius для скругления углов.

Домашнее задание:

- Сделать страницу с различными цветами и градиентами, используя свойства background-color, linear-gradient, radial-gradient и border-radius.
- Сдать HTML-файл или скриншот страницы.

Тема 14: Создание страницы с использованием стилей

Цель лекции:
Научить студентов применять изученные свойства CSS для создания полноценной веб-страницы с оформлением текста, блоков и фоновых элементов.

Применяемые свойства CSS:

- **Текст и заголовки**
- color, font-family, font-size, font-weight, text-align
  Пример:

```
h1 {
  color: navy;
  font-family: Arial, sans-serif;
  font-size: 28px;
  text-align: center;
}
p {
  color: darkslategray;
  font-size: 16px;
}
```

- **Блоки и фон**
- width, height, margin, padding, border, border-radius, background-color, background-image
  Пример:

```
div.container {
  width: 600px;
  margin: 20px auto;
  padding: 20px;
  background-color: lightgray;
  border: 2px solid black;
  border-radius: 10px;
}
```

- **Градиенты и выравнивание элементов**
- linear-gradient, radial-gradient, display, flexbox
  Пример:

```
header {
  background: linear-gradient(to right, orange, yellow);
```

```
  padding: 15px;
  text-align: center;
}

nav {
  display: flex;
  justify-content: space-around;
  background-color: #333;
  color: white;
  padding: 10px;
}
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Создание страницы с CSS</title>
    <style>
      body {
        font-family: Arial, sans-serif;
        background-color: #f0f0f0;
        margin: 0;
      }

      header {
        background: linear-gradient(to right, orange, yellow);
        padding: 15px;
        text-align: center;
        font-size: 24px;
        font-weight: bold;
      }

      nav {
        display: flex;
        justify-content: space-around;
        background-color: #333;
        color: white;
        padding: 10px;
      }

      div.container {
        width: 600px;
        margin: 20px auto;
        padding: 20px;
        background-color: lightgray;
        border: 2px solid black;
```

```
      border-radius: 10px;
    }

    h1 {
      color: navy;
      text-align: center;
    }

    p {
      color: darkslategray;
      font-size: 16px;
    }
  </style>
 </head>
 <body>
  <header>Моя стильная страница</header>

  <nav>
    <div>Главная</div>
    <div>О нас</div>
    <div>Контакты</div>
  </nav>

  <div class="container">
    <h1>Заголовок блока</h1>
    <p>Пример текста внутри блока с фоном, границей и скругленными углами.</p>
    <p>Можно добавить больше абзацев и использовать все стили CSS, которые мы изучили.</p>
  </div>
 </body>
</html>
```

Классная работа:

- Создать страницу с заголовком <header>, навигацией <nav> и блоком с текстом.
- Применить стили для текста, блоков, фоновых цветов и градиентов.
- Использовать flexbox для навигации и скругленные углы для блоков.

Домашнее задание:

- Сделать полноценную страницу с заголовком, навигацией и основным блоком.
- Сдать HTML-файл со всеми стилями или скриншот страницы.

Тема 15: Flexbox: свойства и примеры использования

Цель лекции:
Научить студентов использовать Flexbox для гибкого расположения элементов на странице, выравнивания по горизонтали и вертикали, распределения пространства между элементами.

Основные свойства Flexbox:

- **Контейнер (родительский элемент)**
- display: flex — активирует flex-контейнер
- flex-direction — направление оси (row, column, row-reverse, column-reverse)
- justify-content — выравнивание по основной оси (flex-start, center, flex-end, space-between, space-around)
- align-items — выравнивание по поперечной оси (flex-start, center, flex-end, stretch)
- flex-wrap — перенос элементов (nowrap, wrap, wrap-reverse)

Пример контейнера:

```
div.flex-container {
  display: flex;
  flex-direction: row;
  justify-content: space-between;
  align-items: center;
  flex-wrap: wrap;
  background-color: lightgray;
  padding: 10px;
}
```

- **Элементы внутри Flexbox**
- flex-grow — растягивание элемента (пропорционально)
- flex-shrink — сжатие элемента
- flex-basis — базовый размер элемента
- order — порядок отображения элементов
  Пример элемента:

```
div.flex-item {
  background-color: lightcoral;
  width: 100px;
  height: 50px;
  text-align: center;
  line-height: 50px;
  flex-grow: 1;
}
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
 <head>
  <meta charset="UTF-8">
  <title>Примеры Flexbox</title>
  <style>
   div.flex-container {
    display: flex;
    flex-direction: row;
    justify-content: space-between;
    align-items: center;
    flex-wrap: wrap;
    background-color: lightgray;
    padding: 10px;
   }

   div.flex-item {
    background-color: lightcoral;
    width: 100px;
    height: 50px;
    text-align: center;
    line-height: 50px;
    margin: 5px;
    flex-grow: 1;
   }
  </style>
 </head>
 <body>
  <h1>Примеры Flexbox</h1>

  <div class="flex-container">
   <div class="flex-item">Элемент 1</div>
   <div class="flex-item">Элемент 2</div>
   <div class="flex-item">Элемент 3</div>
   <div class="flex-item">Элемент 4</div>
  </div>
 </body>
</html>
```

Классная работа:

- Создать flex-контейнер с четырьмя элементами.
- Попробовать менять свойства justify-content, align-items, flex-direction, flex-wrap.
- Экспериментировать с flex-grow для элементов.

Домашнее задание:

- Сделать страницу с flex-контейнером и элементами, применяя все изученные свойства Flexbox.
- Сдать HTML-файл или скриншот страницы.

Тема 16: Grid Layout: создание сеток с помощью CSS Grid

Цель лекции:
Научить студентов использовать CSS Grid для создания сеток на веб-странице, управлять расположением элементов в строках и колонках, а также распределением пространства.

Основные свойства CSS Grid:

- **Контейнер (родительский элемент)**
- display: grid — активирует grid-контейнер
- grid-template-columns — задаёт количество и ширину колонок
- grid-template-rows — задаёт количество и высоту строк
- gap — расстояние между элементами (row-gap и column-gap)
  Пример контейнера:

```
div.grid-container {
  display: grid;
  grid-template-columns: repeat(3, 150px);
  grid-template-rows: 100px 100px;
  gap: 10px;
  background-color: lightgray;
  padding: 10px;
}
```

- **Элементы внутри Grid**
- grid-column — определяет, на каких колонках элемент будет располагаться
- grid-row — определяет, на каких строках элемент будет располагаться
  Пример элемента:

```
div.grid-item {
  background-color: lightcoral;
  text-align: center;
  line-height: 100px;
}
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
```

```
  <title>Примеры CSS Grid</title>
  <style>
   div.grid-container {
    display: grid;
    grid-template-columns: repeat(3, 150px);
    grid-template-rows: 100px 100px;
    gap: 10px;
    background-color: lightgray;
    padding: 10px;
   }

   div.grid-item {
    background-color: lightcoral;
    text-align: center;
    line-height: 100px;
   }
  </style>
 </head>
 <body>
  <h1>Примеры CSS Grid</h1>

  <div class="grid-container">
   <div class="grid-item">Элемент 1</div>
   <div class="grid-item">Элемент 2</div>
   <div class="grid-item">Элемент 3</div>
   <div class="grid-item">Элемент 4</div>
   <div class="grid-item">Элемент 5</div>
   <div class="grid-item">Элемент 6</div>
  </div>
 </body>
</html>
```

Классная работа:

- Создать grid-контейнер с шести элементами.
- Попробовать менять количество колонок и строк с помощью grid-template-columns и grid-template-rows.
- Изменять gap и экспериментировать с размерами элементов.

Домашнее задание:

- Сделать страницу с grid-контейнером и элементами, используя свойства CSS Grid.
- Сдать HTML-файл или скриншот страницы.

Тема 17: Псевдоклассы и псевдоэлементы

Цель лекции:
Научить студентов использовать псевдоклассы и псевдоэлементы для изменения внешнего вида элементов в определённых состояниях и для добавления декоративного контента.

Основные понятия:

- **Псевдоклассы** — задают стиль для элемента в определённом состоянии.
  Примеры:
- :hover — когда курсор находится над элементом
- :active — при нажатии на элемент
- :first-child — первый дочерний элемент
- :last-child — последний дочерний элемент

Пример:

```
a:hover {
  color: red;
}

li:first-child {
  font-weight: bold;
}
```

- **Псевдоэлементы** — позволяют стилизовать определённую часть элемента или добавить контент.
  Примеры:
- ::before — вставляет контент перед содержимым элемента
- ::after — вставляет контент после содержимого элемента
- ::first-letter — первый символ текста
- ::first-line — первая строка текста

Пример:

```
p::first-letter {
  font-size: 24px;
  color: blue;
}

div::before {
  content: "Начало блока: ";
  font-weight: bold;
}
```

Пример кода всей страницы

```
<!DOCTYPE html>
```

```
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Псевдоклассы и псевдоэлементы</title>
    <style>
      a {
        color: blue;
        text-decoration: none;
      }

      a:hover {
        color: red;
        text-decoration: underline;
      }

      ul li:first-child {
        font-weight: bold;
      }

      p::first-letter {
        font-size: 24px;
        color: green;
      }

      div.note::before {
        content: "Внимание: ";
        font-weight: bold;
        color: red;
      }
    </style>
  </head>
  <body>
    <h1>Примеры псевдоклассов и псевдоэлементов</h1>

    <a href="#">Ссылка</a>

    <ul>
      <li>Первый элемент списка</li>
      <li>Второй элемент списка</li>
      <li>Третий элемент списка</li>
    </ul>

    <p>Это пример текста с большим первым символом.</p>

    <div class="note">Текст заметки с добавленным псевдоэлементом.</div>
  </body>
</html>
```

Классная работа:

- Создать страницу с ссылкой, списком и абзацем.
- Применить псевдоклассы :hover, :first-child.
- Добавить псевдоэлементы ::before и ::first-letter.

Домашнее задание:

- Сделать страницу, используя псевдоклассы и псевдоэлементы для текста, ссылок и блоков.
- Сдать HTML-файл или скриншот страницы.

Тема 18: Анимации и переходы

Цель лекции:
Научить студентов использовать CSS-переходы и анимации для плавного изменения свойств элементов и создания динамичных эффектов на веб-странице.

Основные свойства:

- **Переходы (transition)** — плавное изменение свойства при изменении состояния элемента.
- transition-property — свойства для анимации
- transition-duration — длительность перехода
- transition-timing-function — скорость изменения (linear, ease, ease-in, ease-out)
  Пример:

```
div.box {
  width: 150px;
  height: 100px;
  background-color: lightblue;
  transition: background-color 0.5s ease, transform 0.5s ease;
}

div.box:hover {
  background-color: lightcoral;
  transform: scale(1.2);
}
```

- **Анимации (animation)** — более сложные движения с ключевыми кадрами.
- @keyframes — определяет последовательность кадров
- animation-name — имя анимации

- animation-duration — длительность
- animation-iteration-count — количество повторений
- animation-timing-function — скорость изменения

Пример:

```
@keyframes moveBox {
  0% { transform: translateX(0); }
  50% { transform: translateX(100px); }
  100% { transform: translateX(0); }
}

div.animated {
  width: 100px;
  height: 100px;
  background-color: lightgreen;
  animation-name: moveBox;
  animation-duration: 2s;
  animation-iteration-count: infinite;
  animation-timing-function: ease-in-out;
}
```

Пример кода всей страницы

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Анимации и переходы</title>
    <style>
      div.box {
        width: 150px;
        height: 100px;
        background-color: lightblue;
        margin-bottom: 20px;
        transition: background-color 0.5s ease, transform 0.5s ease;
      }

      div.box:hover {
        background-color: lightcoral;
        transform: scale(1.2);
      }

      @keyframes moveBox {
        0% { transform: translateX(0); }
        50% { transform: translateX(100px); }
        100% { transform: translateX(0); }
      }
```

```
      div.animated {
        width: 100px;
        height: 100px;
        background-color: lightgreen;
        animation-name: moveBox;
        animation-duration: 2s;
        animation-iteration-count: infinite;
        animation-timing-function: ease-in-out;
      }
    </style>
  </head>
  <body>
    <h1>Примеры анимаций и переходов</h1>

    <div class="box">Наведи на меня</div>

    <div class="animated">Двигаюсь</div>
  </body>
</html>
```

Классная работа:

- Создать блок с эффектом наведения, используя transition.
- Создать блок с непрерывной анимацией движения с помощью @keyframes и animation.

Домашнее задание:

- Сделать страницу с блоками, применяя разные переходы и анимации.
- Сдать HTML-файл или скриншот страницы.

Тема 19: Адаптивная и отзывчивая вёрстка

Цель лекции:
Научить студентов создавать сайты, которые корректно отображаются на разных устройствах и экранах, используя технологию CSS media queries.

Понятия

- **Адаптивная вёрстка** — сайт подстраивается под разные размеры экрана, изменяя расположение элементов.
- **Отзывчивая вёрстка** — сайт плавно подстраивается к любому разрешению (обычно с помощью гибких блоков и процентов).
- **Media Queries** — специальные CSS-условия, которые применяют стили при определённых характеристиках устройства (ширина, высота, ориентация экрана).

Синтаксис media queries

```
@media (условие) {
  /* Стили */
}
```

Примеры условий:

- (max-width: 600px) — стили применяются, если ширина окна ≤ 600px.
- (min-width: 768px) — стили применяются, если ширина окна ≥ 768px.
- (orientation: landscape) — стили применяются, если экран в горизонтальной ориентации.

Пример

```
body {
  font-family: Arial, sans-serif;
  background-color: white;
}

h1 {
  text-align: center;
  color: black;
}

@media (max-width: 600px) {
  body {
    background-color: lightyellow;
  }

  h1 {
    font-size: 18px;
    color: darkblue;
  }
}

@media (min-width: 601px) and (max-width: 1024px) {
  body {
    background-color: lightgreen;
  }

  h1 {
    font-size: 24px;
    color: darkred;
  }
}
```

Полный пример HTML-страницы

```
<!DOCTYPE html>
```

```
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Адаптивная вёрстка</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: white;
    }

    h1 {
      text-align: center;
      color: black;
    }

    @media (max-width: 600px) {
      body {
        background-color: lightyellow;
      }

      h1 {
        font-size: 18px;
        color: darkblue;
      }
    }

    @media (min-width: 601px) and (max-width: 1024px) {
      body {
        background-color: lightgreen;
      }

      h1 {
        font-size: 24px;
        color: darkred;
      }
    }
  </style>
</head>
<body>
  <h1>Пример адаптивной вёрстки</h1>
  <p>Меняй размер окна браузера и смотри, как меняется оформление!</p>
</body>
</html>
```

Классная работа

- Создать страницу, где при ширине экрана до 600px цвет фона меняется на жёлтый, а текст — на синий.

- При ширине от 601px до 1024px — фон зелёный, текст красный.
- При ширине более 1024px — фон белый, текст чёрный.

Домашнее задание

- Сделать страницу с тремя блоками, которые в мобильной версии (до 600px) будут в одну колонку, а на больших экранах — в одну строку.
- Сдать HTML-файл или ссылку на страницу.

Тема 20: Flexbox в адаптивной вёрстке

Цель лекции:
Научить студентов использовать Flexbox для создания гибких адаптивных макетов, которые корректно отображаются на разных размерах экранов.

Основные свойства

- **Контейнер (родительский элемент)**
- display: flex — активирует flex-контейнер
- flex-direction — направление оси (row, column)
- flex-wrap — перенос элементов (wrap, nowrap)
- justify-content — выравнивание по основной оси
- align-items — выравнивание по поперечной оси
- **Элементы внутри Flexbox**
- flex — сокращённая запись для flex-grow, flex-shrink, flex-basis
- order — порядок элементов
- align-self — выравнивание отдельного элемента
- **Адаптивность с media queries**
- Меняем направление (flex-direction) и размеры элементов в зависимости от ширины экрана.

Пример

```
.container {
  display: flex;
  flex-wrap: wrap;
  justify-content: space-between;
  background-color: lightgray;
  padding: 10px;
}

.item {
  background-color: lightcoral;
  width: 200px;
```

```
  height: 100px;
  margin: 5px;
  text-align: center;
  line-height: 100px;
}

/* Адаптивное изменение для мобильных устройств */
@media (max-width: 600px) {
  .container {
    flex-direction: column;
    align-items: center;
  }

  .item {
    width: 90%;
  }
}
```

Полный пример HTML-страницы

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Flexbox в адаптивной вёрстке</title>
  <style>
    .container {
      display: flex;
      flex-wrap: wrap;
      justify-content: space-between;
      background-color: lightgray;
      padding: 10px;
    }

    .item {
      background-color: lightcoral;
      width: 200px;
      height: 100px;
      margin: 5px;
      text-align: center;
      line-height: 100px;
    }

    @media (max-width: 600px) {
      .container {
        flex-direction: column;
        align-items: center;
      }
```

```
      .item {
        width: 90%;
      }
    }
  </style>
</head>
<body>
  <h1>Пример Flexbox в адаптивной вёрстке</h1>

  <div class="container">
    <div class="item">Элемент 1</div>
    <div class="item">Элемент 2</div>
    <div class="item">Элемент 3</div>
    <div class="item">Элемент 4</div>
  </div>
</body>
</html>
```

Классная работа

- Создать flex-контейнер с четырьмя элементами.
- Настроить адаптивное изменение: на мобильных устройствах элементы становятся колонкой, а на больших экранах — строкой.

Домашнее задание

- Сделать страницу с адаптивным макетом из блоков с помощью Flexbox.
- Сдать HTML-файл или скриншот страницы.

Тема 21: Основы SEO для HTML

Цель лекции:
Научить студентов создавать страницы с правильной структурой и метаинформацией для улучшения видимости сайта в поисковых системах.

Основные принципы SEO:

- **Правильная структура HTML**
- Использовать семантические теги (header, nav, main, section, article, footer)
- Заголовки (h1-h6) использовать по иерархии
- Текст должен быть читаемым и структурированным
- **Мета-теги**
- title — заголовок страницы в поисковой выдаче

- meta description — описание страницы, показывается в сниппете
- meta keywords — ключевые слова (сейчас мало используется)

Пример:

```
<head>
  <meta charset="UTF-8">
  <title>Пример SEO страницы</title>
  <meta name="description" content="Пример страницы с правильной SEO структурой.">
  <meta name="keywords" content="HTML, SEO, пример, структура">
</head>
```

- **Альтернативный текст для изображений**
- Атрибут alt помогает поисковым системам понять содержимое изображения

Пример:

```
<img src="logo.png" alt="Логотип компании">
```

- **Чистые URL и семантические ссылки**
- URL должен быть читаемым и описательным
- Использовать ссылки с текстом, а не «кликни здесь»

Пример:

```
<a href="/services.html">Наши услуги</a>
```

- **Внутренние ссылки и навигация**
- Помогают поисковым системам индексировать страницы

Пример кода страницы с SEO

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Пример SEO страницы</title>
  <meta name="description" content="Пример страницы с правильной SEO структурой.">
  <meta name="keywords" content="HTML, SEO, пример, структура">
  <style>
    body { font-family: Arial, sans-serif; margin: 20px; }
    header, nav, main, footer { padding: 10px; margin-bottom: 10px; background-color: #f0f0f0; }
    nav a { margin-right: 10px; text-decoration: none; color: blue; }
  </style>
</head>
<body>
  <header>
    <h1>Пример SEO страницы</h1>
```

```
  </header>

  <nav>
    <a href="index.html">Главная</a>
    <a href="services.html">Услуги</a>
    <a href="contact.html">Контакты</a>
  </nav>

  <main>
    <section>
      <h2>О нас</h2>
      <p>Мы создаём качественные веб-страницы с правильной структурой и
SEO-оптимизацией.</p>
      <img src="team.jpg" alt="Наша команда">
    </section>
    <section>
      <h2>Наши услуги</h2>
      <p>Предоставляем полный спектр услуг по веб-разработке.</p>
    </section>
  </main>

  <footer>
    <p>&copy; 2025 Веб-Студия</p>
  </footer>
</body>
</html>
```

Классная работа

- Создать страницу с заголовками, изображением и навигацией.
- Добавить мета-теги title и description.
- Проверить, чтобы все изображения имели атрибут alt.

Домашнее задание

- Сделать страницу с семантической структурой и SEO-метаинформацией.
- Сдать HTML-файл или ссылку на страницу.

Тема 22: Публикация сайта (GitHub Pages, Netlify)

Цель лекции:
Научить студентов публиковать свои веб-сайты в интернете, используя бесплатные платформы GitHub Pages и Netlify.

Основные понятия:

- **GitHub Pages**
- Позволяет размещать статические сайты прямо из репозитория на GitHub.
- URL формируется по шаблону: https://username.github.io/repository/

Пошаговая инструкция:

- Создать аккаунт на GitHub.
- Создать новый репозиторий и загрузить в него файлы сайта (HTML, CSS, JS).
- Перейти в Settings → Pages.
- Выбрать ветку main и папку /root для публикации.
- Сайт будет доступен по ссылке, указанной в настройках.

- **Netlify**
- Бесплатная платформа для публикации сайтов.
- Поддерживает автоматическое обновление сайта из репозитория (GitHub, GitLab).

Пошаговая инструкция:

- Создать аккаунт на Netlify.
- Нажать New site from Git.
- Выбрать репозиторий с сайтом.
- Настроить ветку и папку для публикации.
- Сайт будет доступен по ссылке вида https://example.netlify.app.

Пример публикации:

- Создаём простой сайт:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Мой первый сайт</title>
  <style>
    body { font-family: Arial, sans-serif; background-color: lightyellow; padding: 20px; }
    h1 { color: darkblue; }
  </style>
</head>
<body>
  <h1>Привет, мир!</h1>
  <p>Это мой первый опубликованный сайт.</p>
```

</body>
</html>

- Загружаем этот сайт на GitHub или Netlify и получаем ссылку для публикации.

Классная работа

- Создать простой HTML-сайт.
- Опубликовать его через GitHub Pages или Netlify.
- Скопировать ссылку и проверить доступность сайта в браузере.

Домашнее задание

- Сделать сайт с несколькими страницами (главная + контакты/услуги).
- Опубликовать на одной из платформ.
- Сдать ссылку на опубликованный сайт.

Практика

Практическое занятие 1: Основы HTML

Цель:
Научить студентов создавать базовую HTML-страницу с заголовками, параграфами и списками.

Задачи занятия:

- Создать HTML-документ с правильной структурой (<!DOCTYPE html>, <html>, <head>, <body>).
- Добавить заголовки разных уровней (h1-h3).
- Добавить параграфы (p).
- Создать маркированный (ul) и нумерованный (ol) списки.

Пример кода
html
КопироватьРедактировать

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Моя первая страница</title>
</head>
<body>
  <h1>Моя первая HTML-страница</h1>
  <h2>Обо мне</h2>
```

```
  <p>Привет! Меня зовут Иван, я учусь создавать веб-страницы.</p>

  <h3>Мои интересы:</h3>
  <ul>
    <li>Программирование</li>
    <li>Музыка</li>
    <li>Спорт</li>
  </ul>

  <h3>План на день:</h3>
  <ol>
    <li>Завтрак</li>
    <li>Учёба</li>
    <li>Прогулка</li>
  </ol>
</body>
</html>
```

Классная работа

- Создать страницу с заголовком и небольшим описанием себя.
- Добавить два списка: один маркированный, один нумерованный.

Домашнее задание

- Создать страницу «Мой профиль», где будет:
    - Заголовок с именем студента.
    - Параграф о себе.
    - Список увлечений (маркированный) и список дел на день (нумерованный).

Практическое занятие 2: Работа со ссылками и изображениями

Цель:
Научить студентов добавлять ссылки и изображения на страницу, использовать основные атрибуты (href, src, alt).

Задачи занятия:

- Добавить внешние и внутренние ссылки с тегом <a>.
- Добавить изображения с тегом <img> и атрибутами src и alt.
- Настроить отображение изображений (ширина, высота, выравнивание).

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
```

```
<head>
  <meta charset="UTF-8">
  <title>Ссылки и изображения</title>
</head>
<body>
  <h1>Пример работы со ссылками и изображениями</h1>

  <h2>Ссылки</h2>
  <p>Внешняя ссылка: <a href="https://example.com" target="_blank">Перейти на Example.com</a></p>
  <p>Внутренняя ссылка: <a href="#section1">Перейти к секции 1</a></p>

  <h2>Изображения</h2>
  <img src="photo.jpg" alt="Фото природы" width="300">
  <img src="logo.png" alt="Логотип сайта" width="150">

  <h2 id="section1">Секция 1</h2>
  <p>Это внутренняя секция, к которой ведет ссылка выше.</p>
</body>
</html>
```

Классная работа

- Создать страницу с:
    - Одной внешней ссылкой на сайт студента (или пример сайта).
    - Одной внутренней ссылкой, которая ведет к секции на странице.
    - Двумя изображениями с атрибутом alt и разной шириной.

Домашнее задание

- Сделать мини-галерею из 3-5 изображений с подписями.
- Добавить ссылки на внешние ресурсы, которые соответствуют интересам студента.

Практическое занятие 3: Семантические теги и структура страницы

Цель:
Научить студентов использовать семантические теги для создания правильной структуры HTML-страницы.

Задачи занятия:

- Использовать теги <header>, <nav>, <main>, <section>, <article>, <footer>.
- Создать логичную и читаемую структуру страницы.
- Добавить ссылки навигации и контент в секции.

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Семантическая страница</title>
</head>
<body>
  <header>
    <h1>Мой сайт</h1>
  </header>

  <nav>
    <a href="#about">О нас</a> |
    <a href="#services">Услуги</a> |
    <a href="#contact">Контакты</a>
  </nav>

  <main>
    <section id="about">
      <h2>О нас</h2>
      <p>Мы создаём качественные веб-страницы с правильной
структурой.</p>
    </section>

    <section id="services">
      <h2>Услуги</h2>
      <article>
        <h3>Веб-разработка</h3>
        <p>Создание сайтов любой сложности.</p>
      </article>
      <article>
        <h3>Дизайн</h3>
        <p>Разработка современного и удобного интерфейса.</p>
      </article>
    </section>
  </main>

  <footer>
    <p>&copy; 2025 Мой сайт</p>
  </footer>
</body>
</html>
```

Классная работа

- Создать страницу с семантической структурой:

- <header> с названием сайта
- <nav> с 2-3 ссылками
- <main> с 2 секциями
- <footer> с копирайтом

Домашнее задание

- Создать страницу «Моя мини-компания» с:
  - Заголовком и навигацией
  - 2-3 секциями с контентом (текст + изображение)
  - Семантическими тегами для каждой части страницы

Практическое занятие 4: Таблицы в HTML

Цель:
Научить студентов создавать таблицы, использовать строки и ячейки, а также основные атрибуты для оформления таблиц.

Задачи занятия:

- Создать таблицу с заголовками (<th>) и данными (<td>).
- Использовать атрибуты таблицы: border, cellpadding, cellspacing.
- Добавить несколько строк и столбцов, чтобы таблица была информативной.

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Пример таблицы</title>
</head>
<body>
  <h1>Расписание занятий</h1>

  <table border="1" cellpadding="5" cellspacing="0">
    <tr>
      <th>День недели</th>
      <th>Предмет</th>
      <th>Время</th>
    </tr>
    <tr>
      <td>Понедельник</td>
      <td>Математика</td>
```

```
    <td>09:00 - 10:30</td>
   </tr>
   <tr>
    <td>Вторник</td>
    <td>Физика</td>
    <td>10:45 - 12:15</td>
   </tr>
   <tr>
    <td>Среда</td>
    <td>Информатика</td>
    <td>13:00 - 14:30</td>
   </tr>
  </table>
</body>
</html>
```

Классная работа

- Создать таблицу «Расписание студентов» с 3-4 строками и 3 столбцами.
- Использовать атрибуты border, cellpadding и cellspacing.

Домашнее задание

- Добавить в таблицу ещё 2 дня недели и новые предметы.
- Настроить ширину столбцов с помощью CSS (style="width:...") для улучшения визуального вида.

Практическое занятие 5: Создание и стилизация пользовательских форм

Цель:
Научить студентов создавать формы для ввода данных и применять базовую стилизацию с помощью HTML и CSS.

Задачи занятия:

- Создать форму с различными полями ввода (<input>), текстовой областью (<textarea>) и выпадающим списком (<select>).
- Добавить кнопку отправки (<button>).
- Использовать атрибуты формы: action, method, type, name, placeholder.
- Применить базовые стили CSS для улучшения внешнего вида формы.

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
```

```
  <meta charset="UTF-8">
  <title>Форма обратной связи</title>
  <style>
    form {
      max-width: 400px;
      margin: 20px auto;
      padding: 15px;
      border: 1px solid #333;
      background-color: #f9f9f9;
    }
    input, textarea, select, button {
      width: 100%;
      margin-bottom: 10px;
      padding: 8px;
      font-size: 14px;
    }
  </style>
</head>
<body>
  <h1>Форма обратной связи</h1>
  <form action="/submit" method="post">
    <input type="text" name="username" placeholder="Ваше имя">
    <input type="email" name="email" placeholder="Ваш email">
    <textarea name="message" placeholder="Сообщение"></textarea>
    <select name="subject">
      <option>Общий вопрос</option>
      <option>Поддержка</option>
      <option>Предложение</option>
    </select>
    <button type="submit">Отправить</button>
  </form>
</body>
</html>
```

Классная работа

- Создать форму «Обратная связь» с полями: Имя, Email, Сообщение, Тема.
- Добавить кнопку «Отправить».
- Применить простые стили CSS для улучшения внешнего вида формы.

Домашнее задание

- Добавить новые поля в форму: Телефон, Дата рождения.
- Изменить цвет фона и рамки формы с помощью CSS.
- Проверить работу формы в браузере.

Практическое занятие 6: Подключение CSS и работа с основными свойствами

Цель:
Научить студентов подключать CSS к HTML и использовать основные свойства для стилизации элементов.

Задачи занятия:

- Подключить CSS к HTML тремя способами:
  - Inline (style="")
  - Internal (<style> в <head>)
  - External (внешний файл .css)
- Изучить и применить основные свойства:
  - Цвета (color, background-color)
  - Шрифты (font-family, font-size, font-weight)
  - Размеры и отступы (width, height, margin, padding)
  - Границы (border)

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
 <meta charset="UTF-8">
 <title>Пример CSS</title>
 <style>
  body {
   font-family: Arial, sans-serif;
   background-color: #f0f0f0;
   padding: 20px;
  }
  h1 {
   color: darkblue;
   font-size: 28px;
  }
  p {
   color: #333;
   font-size: 16px;
   margin-bottom: 10px;
  }
  .box {
   width: 200px;
   height: 100px;
   background-color: lightgreen;
   border: 2px solid green;
```

```
    padding: 10px;
    margin-bottom: 10px;
  }
 </style>
</head>
<body>
 <h1>Мой первый стильный блок</h1>
 <p>Это пример текста с использованием CSS.</p>
 <div class="box">Контейнер с цветом и рамкой</div>
</body>
</html>
```

Классная работа

- Создать страницу с заголовком и параграфом.
- Добавить блок (div) с цветным фоном и рамкой.
- Применить CSS через Internal и Inline стили.

Домашнее задание

- Создать внешний файл CSS и подключить его к странице.
- Изменить шрифты, цвета, размеры и отступы текста и блоков.
- Сделать несколько блоков с разными цветами и рамками.

Практическое занятие 7: Блочная модель и позиционирование элементов

Цель:
Научить студентов понимать и использовать блочную модель CSS для стилизации элементов, а также позиционировать блоки на странице.

Задачи занятия:

- Изучить блочную модель: margin, padding, border, width, height.
- Использовать свойства позиционирования: position (static, relative, absolute, fixed), float.
- Создать визуально аккуратные блоки с отступами и рамками.

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
 <meta charset="UTF-8">
 <title>Блочная модель и позиционирование</title>
 <style>
  .block {
```

```
    width: 200px;
    height: 100px;
    margin: 20px; /* внешние отступы */
    padding: 10px; /* внутренние отступы */
    border: 2px solid #333; /* рамка */
    background-color: lightblue;
   }

   .relative-block {
    position: relative;
    top: 20px;
    left: 20px;
    background-color: lightgreen;
   }

   .absolute-block {
    position: absolute;
    top: 150px;
    left: 50px;
    background-color: lightcoral;
   }

   .float-block {
    float: right;
    width: 100px;
    height: 100px;
    background-color: yellow;
    margin: 10px;
   }
  </style>
</head>
<body>
  <h1>Пример блочной модели</h1>

  <div class="block">Простой блок</div>
  <div class="relative-block">Относительное позиционирование</div>
  <div class="absolute-block">Абсолютное позиционирование</div>
  <div class="float-block">Плавающий блок</div>
</body>
</html>
```

Классная работа

- Создать три блока с разными цветами и размерами.
- Применить margin, padding и border.
- Использовать relative, absolute и float для позиционирования блоков на странице.

Домашнее задание

- Сделать страницу с 4 блоками, каждый из которых использует разные типы позиционирования.
- Настроить внешние и внутренние отступы, а также ширину и высоту блоков.

Практическое занятие 8: Flexbox на практике

Цель:
Научить студентов использовать Flexbox для создания адаптивных и аккуратно выровненных блоков на странице.

Задачи занятия:

- Изучить контейнер Flex: display: flex.
- Управлять расположением элементов с помощью justify-content, align-items, flex-direction и flex-wrap.
- Создать гибкую структуру блоков, которая меняет расположение при изменении ширины экрана.

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Flexbox пример</title>
  <style>
    .flex-container {
      display: flex;
      justify-content: space-around; /* распределение по горизонтали */
      align-items: center; /* выравнивание по вертикали */
      flex-wrap: wrap; /* перенос на следующую строку при нехватке места */
      background-color: #f0f0f0;
      padding: 20px;
    }

    .flex-item {
      width: 150px;
      height: 100px;
      background-color: lightblue;
      border: 2px solid #333;
      margin: 10px;
      text-align: center;
      line-height: 100px;
```

```
  }
 </style>
</head>
<body>
 <h1>Пример Flexbox</h1>
 <div class="flex-container">
  <div class="flex-item">Блок 1</div>
  <div class="flex-item">Блок 2</div>
  <div class="flex-item">Блок 3</div>
  <div class="flex-item">Блок 4</div>
 </div>
</body>
</html>
```

Классная работа

- Создать контейнер с 4-5 блоками.
- Использовать justify-content и align-items для распределения элементов.
- Сделать так, чтобы блоки переносились на новую строку при уменьшении ширины окна.

Домашнее задание

- Создать адаптивный ряд карточек (минимум 4), который меняет направление с row на column на узком экране.
- Настроить отступы и размеры блоков с помощью margin и width.

Практическое занятие 9: Grid Layout на практике

Цель:
Научить студентов создавать сетки с помощью CSS Grid для аккуратного распределения элементов на странице.

Задачи занятия:

- Создать контейнер Grid: display: grid.
- Настроить количество колонок и строк с помощью grid-template-columns и grid-template-rows.
- Добавить промежутки между элементами с помощью grid-gap.
- Размещать элементы на сетке и управлять их размером.

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
 <meta charset="UTF-8">
```

```
  <title>Grid Layout пример</title>
  <style>
    .grid-container {
      display: grid;
      grid-template-columns: repeat(3, 1fr); /* 3 колонки одинаковой ширины */
      grid-gap: 10px; /* промежутки между блоками */
      background-color: #f0f0f0;
      padding: 20px;
    }

    .grid-item {
      background-color: lightcoral;
      border: 2px solid #333;
      height: 100px;
      text-align: center;
      line-height: 100px;
      font-weight: bold;
      color: white;
    }
  </style>
</head>
<body>
  <h1>Пример CSS Grid</h1>
  <div class="grid-container">
    <div class="grid-item">Блок 1</div>
    <div class="grid-item">Блок 2</div>
    <div class="grid-item">Блок 3</div>
    <div class="grid-item">Блок 4</div>
    <div class="grid-item">Блок 5</div>
    <div class="grid-item">Блок 6</div>
  </div>
</body>
</html>
```

Классная работа

- Создать сетку из 6-9 блоков.
- Использовать grid-template-columns и grid-gap для выравнивания.
- Сделать сетку адаптивной: на узких экранах 1-2 колонки.

Домашнее задание

- Создать страницу «Портфолио» с сеткой из 6 карточек (изображение + подпись).
- Настроить адаптивность с помощью grid-template-columns и медиа-запросов.

Практическое занятие 10: Анимации и переходы

Цель:
Научить студентов создавать анимацию и плавные переходы элементов с помощью CSS.

Задачи занятия:

- Использовать CSS-переходы (transition) для плавного изменения свойств.
- Создавать простые анимации с помощью @keyframes и animation.
- Применять анимации к блокам и тексту.

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Пример анимаций</title>
  <style>
    .transition-box {
      width: 150px;
      height: 100px;
      background-color: lightblue;
      margin: 20px;
      transition: background-color 0.5s, transform 0.5s;
    }

    .transition-box:hover {
      background-color: lightcoral;
      transform: scale(1.2);
    }

    .animation-box {
      width: 150px;
      height: 100px;
      background-color: lightgreen;
      margin: 20px;
      animation: moveBox 3s infinite alternate;
    }

    @keyframes moveBox {
      0% { transform: translateX(0); }
      100% { transform: translateX(200px); }
    }
  </style>
</head>
```

```
<body>
  <h1>Пример анимаций и переходов</h1>

  <div class="transition-box">Наведи на меня</div>
  <div class="animation-box">Я двигаюсь</div>
</body>
</html>
```

Классная работа

- Создать блок с переходом цвета и увеличением при наведении.
- Создать блок с анимацией движения, вращения или изменения размера.

Домашнее задание

- Создать страницу с 2-3 блоками, каждый из которых имеет свою анимацию.
- Настроить длительность, задержку и количество повторений анимации с помощью CSS.

Практическое занятие 11: Адаптивная вёрстка с медиа-запросами

Цель:
Научить студентов создавать страницы, которые корректно отображаются на разных устройствах и экранах.

Задачи занятия:

- Изучить и применять медиа-запросы (@media).
- Настроить изменения стилей для разных ширин экрана.
- Сделать элементы страницы гибкими и адаптивными.

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Адаптивная страница</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
    }

    .container {
```

```
    display: flex;
    flex-direction: row;
    justify-content: space-around;
    padding: 20px;
  }

  .box {
    width: 30%;
    height: 100px;
    background-color: lightblue;
    text-align: center;
    line-height: 100px;
    margin: 10px;
    border: 2px solid #333;
  }

  /* Медиа-запрос для экранов шириной менее 600px */
  @media (max-width: 600px) {
    .container {
      flex-direction: column;
      align-items: center;
    }

    .box {
      width: 80%;
    }
  }
 </style>
</head>
<body>
 <h1>Пример адаптивной страницы</h1>
 <div class="container">
  <div class="box">Блок 1</div>
  <div class="box">Блок 2</div>
  <div class="box">Блок 3</div>
 </div>
</body>
</html>
```

Классная работа

- Создать три блока в ряду для больших экранов.
- Настроить медиа-запрос: на узких экранах блоки должны располагаться вертикально.

Домашнее задание

- Создать адаптивную страницу с 4 блоками.

- Настроить разные стили текста и блоков для экранов шириной меньше 800px и 500px.
- Проверить работу страницы на мобильном устройстве или уменьшении окна браузера.

Практическое занятие 12: Итоговый проект

Цель:
Применить все изученные знания HTML и CSS для создания полноценной веб-страницы.

Задачи занятия:

- Создать семантическую структуру страницы с <header>, <nav>, <main>, <section>, <article>, <footer>.
- Использовать таблицы, формы, списки, изображения и ссылки.
- Применить CSS для стилизации текста, блоков, таблиц и форм.
- Сделать страницу адаптивной с помощью Flexbox, Grid и медиа-запросов.
- Добавить простые анимации или переходы для интерактивности.

Пример кода

```
<!DOCTYPE html>
<html lang="ru">
<head>
 <meta charset="UTF-8">
 <title>Мой итоговый проект</title>
 <style>
  body {
   font-family: Arial, sans-serif;
   margin: 0;
   padding: 0;
   background-color: #f4f4f4;
  }
  header, footer {
   background-color: #333;
   color: white;
   text-align: center;
   padding: 15px;
  }
  nav a {
   margin: 0 10px;
   color: white;
   text-decoration: none;
  }
```

```
    main {
      padding: 20px;
    }
    .grid-container {
      display: grid;
      grid-template-columns: repeat(3, 1fr);
      grid-gap: 10px;
      margin-bottom: 20px;
    }
    .grid-item {
      background-color: lightblue;
      text-align: center;
      padding: 20px;
      border: 2px solid #333;
    }
    form input, form textarea, form button {
      width: 100%;
      margin-bottom: 10px;
      padding: 8px;
    }
    @media (max-width: 600px) {
      .grid-container {
        grid-template-columns: 1fr;
      }
    }
  </style>
</head>
<body>
  <header>
    <h1>Мой проект</h1>
    <nav>
      <a href="#about">О нас</a>
      <a href="#services">Услуги</a>
      <a href="#contact">Контакты</a>
    </nav>
  </header>

  <main>
    <section id="about">
      <h2>О нас</h2>
      <p>Мы создаем качественные веб-страницы с правильной
структурой.</p>
    </section>

    <section id="services">
      <h2>Услуги</h2>
      <div class="grid-container">
```

```
      <div class="grid-item">Веб-разработка</div>
      <div class="grid-item">Дизайн</div>
      <div class="grid-item">SEO</div>
    </div>
  </section>

  <section id="contact">
    <h2>Свяжитесь с нами</h2>
    <form action="/submit" method="post">
      <input type="text" name="name" placeholder="Ваше имя">
      <input type="email" name="email" placeholder="Ваш email">
      <textarea name="message" placeholder="Сообщение"></textarea>
      <button type="submit">Отправить</button>
    </form>
  </section>
 </main>

 <footer>
  <p>&copy; 2025 Мой проект</p>
 </footer>
</body>
</html>
```

Классная работа

- Создать страницу «Моя мини-компания» с:
    - Заголовком и навигацией
    - 2-3 секциями с контентом и изображениями
    - Формой обратной связи
    - Адаптивной сеткой и стилизованными блоками

Домашнее задание

- Доработать страницу:
    - Добавить таблицу с расписанием или услугами
    - Применить анимации или переходы к элементам
    - Проверить адаптивность на разных экранах

Рекомендации по работе с литературой

При изучении курса учебной дисциплины особое внимание следует обратить на следующие литературные источники:

1. Ловери, Джозеф, В Dreamweaver MX. Библия пользователя. : Пер. с англ. – М.: Издательский дом «Вильямс», 2003. – 1296 с.
2. Астафурова О.А. Основы Web-технологии. – Волгоград:ВАГС, 2010
3. Круг. С. Веб-дизайн - Пер. с англ. - СПб: Символ-Плюс, 2001.
4. Кото К. и Котлер Э. Веб-редизайн: - Пер. с англ. - СПб: Символ-Плюс, 2003.

Разъяснения по поводу работы с тестовой системой курса

В разделе 1 настоящего комплекса Вам предложено решить тестовые задания. Специфика решения данных заданий заключается в необходимости найти правильный вариант ответа из четырех предложенных.

Советы по подготовке к экзамену (зачету)

При подготовке к зачету особое внимание следует обратить на следующие моменты: необходимо уметь выполнять практические задания по всем темам курса и обладать соответствующими теоретическими знаниями.

Словарь основных терминов (глоссарий)

Web-страница –- документ, который можно открыть и посмотреть с помощью программы просмотра – браузера.

Web-сайт – совокупность Web-страниц, объединенных по смыслу и навигационно.

Web-сервер – компьютер, подключенный к сети, на котором хранятся Web-страницы и Web-сайты.

World Wide Web (WWW, Всемирная паутина) – это наиболее популярный вид информационных услуг Internet, основанный на архитектуре клиент-сервер.

HTML (HyperText Markup Language) – гипертекстовый язык разметки.

HTML-теги – инструкции (команды) по форматированию документа, написанного на языке HTML.

Атрибуты – инструкции уточняющие действие тега.

Web-редактор – программа, предназначенная для создания Web-сайта.

Сценарий –программный код, обеспечивающий интерактивность Web-страниц.

Браузеры – программа, предназначенная для просмотра Web-страниц.

Материалы текущего, промежуточного и итогового контроля

Вопросы для проведения внутри семестровой аттестации:

1. Определения: Web-дизайн, Web-страница и Web-сайт, Web-сервер.
2. История развития World Wide Web
3. Язык разметки.
4. Обзор Web-редакторов.
5. HTML-код, HTML-теги и атрибуты.
6. Структура HTML-кода
7. Технология "клиент-сервер".
8. Обзор Браузеров.
9. Поисковые системы

Вопросы к зачету:

1. Редакторы для создания Web-страниц.
2. Какие теги должны присутствовать в HTML-документе обязательно?
3. Какова логическая структура Web-страницы?
4. Задание свойств страницы.
5. В каких случаях удобнее использовать в гиперссылках абсолютные, а каких – относительные адреса?
6. Добавлением подключение нестандартных шрифтов.
7. Вставка изображений.
8. Вставка фоновых изображений.
9. Вставка рекламных баннеров.
10. Установка ссылок: на файлы, на адреса электронной почты, на закладки.
11. Вставка таблицы. свойства таблиц.
12. Вставка формы.
13. Виды полей формы.
14. Создание списков.
15. Создание слоев.
16. Добавление сценариев.
17. Аттестация Web-страниц.

Список литературы

Основная литература:

1) Гончаров А.Ю. Web-дизайн: HTML, JavaScript и CSS. Карманный справочник.
2) Раджабов К.Я. Учебно-методический комплекс по дисциплине
3) "Web-программирование". Махачкала: ДГИНХ. 2006, 174с.Кото К. и Котлер Э. Веб-редизайн: - Пер. с англ. - СПб: Символ-Плюс, 2003.
4) Дронов В. Профессиональное программирование. HTML 5, CSS 3 и Web 2.0. Разработка современных Web-сайтов. BHV-СПб, 2011 г., 416
5) c. dgunh.ru/content/glavnay/ucheb_deyatel/uposob/up-it_ib-fgos-41.pdf

Материалы текущего, промежуточного и итогового контроля
Примерный тест промежуточного контроля по дисциплине

Тест

1) Какой тег задаёт структуру HTML-документа и указывает тип файла?
    a) <head>
    b) <title>
    c) <!DOCTYPE html>
    d) <body>

2) Какой атрибут тега <html> определяет язык документа?
    a) charset
    b) name
    c) lang
    d) type

3) Какой тег используется для добавления изображения?
    a) <img>
    b) <pic>
    c) <image>
    d) <photo>

4) Какой атрибут изображения отвечает за альтернативный текст?
    a) title
    b) alt
    c) src
    d) desc

5) Какой тег создаёт маркированный список?
    a)
    b)
    c)
    d) <list>

6) Какой тег используется для создания ссылки?
    a) <link>
    b) <a>
    c) <href>
    d) <src>

7) Какой CSS-свойство задаёт цвет текста?
    a) background-color
    b) border-color
    c) color
    d) text-align

8) Какой способ подключения CSS является встроенным (inline)?
   a) <link rel="stylesheet" href="style.css">
   b) <style></style>
   c) style="color:red"
   d) @import url();

9) Что делает свойство display: flex;?
   a) Удаляет элемент
   b) Делает элемент прозрачным
   c) Создаёт flex-контейнер
   d) Добавляет тень

10) Какое свойство Flexbox меняет направление элементов?
   a) flex-wrap
   b) justify-content
   c) flex-direction
   d) align-items

11) Какой тег используется для вставки аудио?
   a) <sound>
   b) <audio>
   c) <mp3>
   d) <music>

12) Какой атрибут отвечает за автоматическое воспроизведение видео/аудио?
   a) auto
   b) play
   c) autoplay
   d) loop

13) Какой мета-тег отвечает за описание страницы для поисковиков?
   a) <meta name="keywords">
   b) <meta name="author">
   c) <meta name="description">
   d) <meta charset="UTF-8">

14) Какая платформа позволяет публиковать сайт прямо из GitHub-репозитория?
   a) XAMPP
   b) Netlify
   c) Visual Studio Code
   d) Photoshop

15) Какой тег задаёт основную структуру разделов HTML5 (основной контент)?

a) <main>
b)
c) <content>
d) <article>

Правильные ответы

1 — c
2 — c
3 — a
4 — b
5 — b
6 — b
7 — c
8 — c
9 — c
10 — c
11 — b
12 — c
13 — c
14 — b
15 — a

1) Какой тег HTML используется для создания формы?
   a) <input>
   b) <form>
   c) <button>
   d) <label>

2) Какой атрибут определяет адрес, куда отправляются данные формы?
   a) action
   b) method
   c) target
   d) name

3) Какой метод отправки формы передает данные в адресной строке?
   a) PUT
   b) POST
   c) GET
   d) SEND

4) Какой тип input используется для выбора одного варианта из нескольких?
   a) checkbox
   b) radio

c) text
d) select

5) Какое CSS-свойство задаёт внешний отступ элемента?
   a) padding
   b) border
   c) margin
   d) gap

6) Что делает свойство transition в CSS?
   a) Меняет фон
   b) Создаёт сетку
   c) Добавляет анимацию плавного изменения
   d) Устанавливает высоту элемента

7) Какое свойство отвечает за задержку начала анимации?
   a) animation-name
   b) animation-delay
   c) animation-speed
   d) animation-start

8) Какое свойство CSS Grid задаёт количество колонок?
   a) grid-rows
   b) grid-template-columns
   c) flex-direction
   d) columns

9) Какое CSS свойство отвечает за размещение элементов по горизонтали в Flexbox?
   a) align-items
   b) justify-content
   c) flex-flow
   d) position

10) Какой мета-тег используется для адаптивной вёрстки на телефонах?
   a) <meta name="screen">
   b) <meta name="responsive">
   c) <meta name="viewport">
   d) <meta name="mobile">

11) Какое расширение обычно имеют файлы каскадных таблиц?
   a) .html
   b) .css
   c) .js
   d) .txt

12) Какой сервис используется для публикации сайта через GitHub Pages?
   a) GitHub Settings → Pages
   b) Photoshop Export
   c) Chrome Developer Tools
   d) MySQL

13) Какой атрибут используется у <img> для указания файла изображения?
   a) alt
   b) src
   c) path
   d) href

14) Какой элемент HTML используется для встроенного видео?
   a) <video>
   b) <media>
   c) <movie>
   d) <clip>

15) Что означает SEO?
   a) Social Engine Operation
   b) Site Earning Optimization
   c) Search Engine Optimization
   d) System Environment Options

Правильные ответы (1–15)

1 — b
2 — a
3 — c
4 — b
5 — c
6 — c
7 — b
8 — b
9 — b
10 — c
11 — b
12 — a
13 — b
14 — a
15 — c