Федеральное государственное бюджетное образовательное учреждение
высшего образования
Казанский государственный энергетический университет
Кафедра информационных технологий и интеллектуальных систем

Введение в HTML и CSS

Методические указания для выполнения лабораторных работ

Казань, 2024

Содержание

1 Введение в Web-технологии и язык HTML

1.1 Основные понятия Web-технологий

Рассмотрим основную терминологию Web-технологий.

Web-технология - набор программ для обеспечения взаимодействия клиент-сервер в сетях Internet или Intranet.

Сервер (**Server** от serve — служить) в информационных технологиях — аппаратный или программный компонент вычислительной системы, выполняющий специализированные функции по запросу клиента, предоставляя ему доступ к определенным ресурсам. Сервер, реализованный в виде программы или программного модуля, обычно выполняет строго определённую задачу и обменивается информацией с клиентом по заранее определённому протоколу. Примерами программных серверов могут служить: файл-сервер, сервер печати, веб-сервер (Apache, IIS), сервер БД, почтовый сервер (Sendmail Postfix), OLE-сервер, ActiveX-сервер и т. п.

Клиент — (в информационных технологиях) это аппаратный или программный компонент вычислительной системы, посылающий запросы серверу. Программа, являющаяся клиентом, взаимодействует с сервером, используя определенный протокол. Она может запрашивать с сервера какие-либо данные, манипулировать данными непосредственно на сервере, запускать на сервере новые процессы и т. п. Полученные от сервера данные клиентская программа может предоставлять пользователю или использовать как-либо иначе, в зависимости от назначения программы. Программа-клиент и программа-сервер могут работать как на одном и том же компьютере, так и на разных.

Интернет (**Internet**, сокращённое от INTERconnected NETworks – объединённые сети; сленговое инéт, нет) – глобальная всемирная телекоммуникационная сеть, обеспечивающая связь для пересылки сообщений электронной почты, передачи файлов, соединения с другими компьютерами и получения доступа к информации, существующей в самых различных формах.

Интранет (Intranet) – корпоративная - локальная или территориально распределенная сеть, закрытая от внешнего доступа из Интернет. Такая сеть, возможно, использует публичные каналы связи, входящие в Интернет, но при этом обеспечивается защита передаваемых данных и меры по пресечению проникновения извне на корпоративные узлы.

WWW (World Wide Web - «всемирная паутина») - глобальное информационное пространство, основанное на физической инфраструктуре Интернета и протоколе передачи данных HTTP.

Понятие WWW часто путают с понятием Интернет. Интернет состоит из огромного количества компьютеров и сетей, в то время как всемирную паутину составляет множество вебсайтов. Помимо WWW посредством Интернета работает множество различных служб: e-mail, IP-телефония, Интернет-радио и телевидение, файловые серверы, компьютерные игры и др.

Вся технология WWW основана на четырех основных элементах:

- язык гипертекстовой разметки документов HTML (Hyper Text Markup Language);
- универсальный способ адресации ресурсов в сети URL (Universal Resource Locator);
- протокол обмена гипертекстовой информацией HTTP (Hyper Text Transfer Protocol);
- универсальный интерфейс шлюзов CGI (Common Gateway Interface).

HTML (HyperText Markup Language — «язык гипертекстовой разметки») — стандартный язык разметки документов во Всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML. Язык HTML интерпретируется браузерами и отображается в виде документа в удобной для человека форме.

URL (Uniform Resource Locator) - единый указатель ресурсов в сети Интернет. URL служит стандартизированным способом записи адреса ресурса в Интернет. Структура URL:

протокол :// пользователь : пароль @ хост : порт / путь ? запрос

При помощи URL можно ссылаться на любой открытый ресурс, будь то страница, изображение, файл для загрузки и т.д.

Для обращения к сайту используют **доменное имя.** Доменное имя состоит из нескольких доменов, разделенных точкой.

Например:

maps.google.com - означает, что домен третьего уровня maps входит в домен второго уровня google, который в свою очередь входит в домен первого (верхнего) уровня com.

Домены верхнего уровня делятся на общие и географические. Общие домены предназначены для определенного класса организаций, например:

- .com - для коммерческих сайтов;
- .org - для некоммерческих организаций;
- .net - для сайтов, чья деятельность связана с Интернетом.

Географические домены выделяются для конкретной страны, например: .ru - Россия; .us – США; .eu - Европейский союз; .de – Германия.

Помимо доменного имени для работы сайта необходим круглосуточно работающий веб-сервер, способный выдержать большие нагрузки. Услуга размещения веб-сайта на веб-сервере называется **хостингом.** Хостинг может быть платным или бесплатным. На платном хостинге, как правило, предоставляется доменное имя второго уровня. За пользование хостингом и доменом взимается абонентская плата. На бесплатном хостинге предоставляется доменное имя третьего уровня (напр. example.narod.ru). По сравнению с платным хостингом, бесплатный имеет ограниченную функциональность, возможен принудительный показ рекламы на размещаемом сайте и т.п.

HTTP (Hypertext Transfer Protocol - «протокол передачи гипертекста») - предназначен для установления связи с веб-сервером и обеспечения доставки

HTML-страниц веб-браузеру клиента. Иначе говоря, HTTP - это «язык», на котором общаются браузер и сервер.

Гипертекст - размеченный текст, содержащий в себе ссылки на внешние ресурсы. Термин гипертéкст был введён Тедом Нельсоном в 1965 году для обозначения «текста ветвящегося или выполняющего действия по запросу». Обычно гипертекст представляется набором текстов, содержащих узлы перехода от одного текста к какому-либо другому, позволяющие избирать читаемые сведения или последовательность чтения. Общеизвестным и ярко выраженным примером гипертекста служат веб-страницы — документы HTML (язык разметки гипертекста), размещённые в сети. В более широком понимании термина, гипертекстом является любая повесть, словарь или энциклопедия, где встречаются отсылки к другим частям данного текста.

В компьютерной терминологии, гипертекст — текст, сформированный с помощью языка разметки, потенциально содержащий в себе ссылки.

Гиперссылка – это объект Web-страницы, содержащий информацию об адресе другой Web-страницы. В качестве такого объекта обычно выступает фрагмент текста, выделенный цветом и подчеркиванием, или графическая иллюстрация, выделенная цветной рамкой.

Гипертекстовые ссылки обычно «указывают» на Web-страницу, тематически связанную со страницей, просматриваемой в данный момент. Текст, являющийся ссылкой, может описывать содержание нового документа. При наведении на гиперссылку указатель мыши принимает форму кисти руки с вытянутым указательным пальцем.

Web-страница - гипертекстовой ресурс Всемирной паутины, обычно написанный на языке HTML. Веб-страница может содержать ссылки для перехода на другие страницы, а также изображения, медиафайлы, например звуковые файлы и видео, Flash-анимацию и т.п.

Web-сайт - несколько web-страниц, объединенных общей темой, дизайном, а также связанных между собой ссылками и обычно находящихся на одном и том же web-сервере. Для загрузки и просмотра web-страниц используются специальные программы — браузеры.

CGI (Common Gateway Interface — «общий интерфейс шлюза») — стандарт интерфейса, используемого для связи внешней программы с веб-сервером. Программу, которая работает по такому интерфейсу совместно с веб-сервером, принято называть шлюзом, хотя многие предпочитают названия «скрипт» (сценарий) или «CGI-программа».

Сам интерфейс разработан таким образом, чтобы можно было использовать любой язык программирования, который может работать со стандартными устройствами ввода-вывода. Такими возможностями обладают даже скрипты для встроенных командных интерпретаторов операционных систем, поэтому в простых случаях могут использоваться даже командные скрипты.

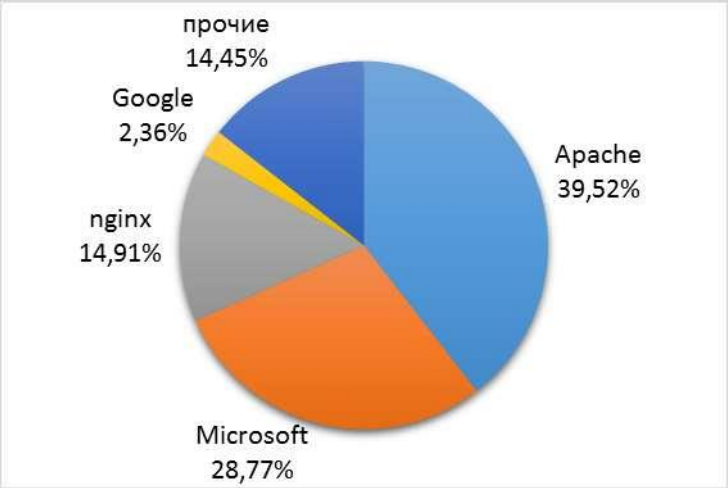
CGI является одним из наиболее распространённых средств создания динамических веб-страниц.

1.2 Программное обеспечение WWW

Веб-серверы

Веб-сервер — это сетевое приложение, обслуживающее HTTP-запросы от клиентов, обычно веб-браузеров. Веб-сервер принимает запросы и возвращает ответы, обычно вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными. Веб-серверы — основа Всемирной паутины.

Созданием программного обеспечения web-серверов занимаются многие разработчики, но наибольшую популярность в WWW получили такие программные продукты, как Apache (Apache Software Foundation), IIS (Microsoft), nginx (И. Сысоев) и Google Web Server (GWS, Google Inc.) (рисунок 1).



Web-сервер	Количество сайтов	Доля, %
Apache	337,175,536	39,52
Microsoft IIS	245,496,533	28,77
nginx	127,191,696	14,91
Google	20,097,702	2,36
прочие	123,290,492	14,45
Итого	853,251,959	100,00

Рисунок 1 - Распределение web-серверов на март 2015 г. (по данным netcraft.com)

Apache - свободный web-сервер с открытым исходным кодом, распространяется под совместимой с GPL лицензией. Apacheуже многие годы является лидером по распространенности во Всемирной паутине в силу своей надежности, гибкости, масштабируемости и безопасности.

IIS (Internet Information Services) - проприетарный набор серверов для нескольких служб Интернета, разработанный Майкрософт и распространяемый с ОС семейства Windows NT. Основным компонентом IIS является веб-сервер, также поддерживаются протоколы FTP, POP3, SMTP, NNTP.

Nginx [enginex] - это HTTP-сервер, совмещенный с почтовым прокси-сервером. Разработан И. Сысоевым для компании Рамблер. Осенью 2004 года

вышел первый публично доступный релиз, сейчас nginx используется рядом крупных сайтов.

Google WebServer (GWS) - разработка компании Googlena основе web-сервера Apache.GWSоптимизирован для выполнения приложений сервиса GoogleApplications.

QZHTTP - модифицированный Apache, используемый на китайском портале qq.com. На нем размещены сервисы онлайн-дневников и мгновенного обмена сообщениями.

Lighttpd - веб-сервер, разрабатываемый с расчётом на быстроту и защищённость при использовании на сильно нагруженных сайтах, а также соответствие стандартам. Lighttpd— свободное программное обеспечение, распространяемое по лицензии BSD.

Браузеры

Браузер, веб-обозреватель (web-browser) — это клиентское программное обеспечение для доступа к веб-серверам по протоколу HTTP и просмотра веб-страниц. Как правило браузеры дополнительно поддерживают и ряд других протоколов (например, ftp, file, mms, pop3).

Первые HTTP-клиенты были консольными и работали в текстовом режиме, позволяя читать гипертекст и перемещаться по ссылкам. Сейчас консольные браузеры (такие, как lynx, w3m или links) практически не используются рядовыми посетителями веб-сайтов. Тем не менее такие браузеры весьма полезны для веб-разработчиков, так как позволяют "увидеть" веб-страницу "глазами" поискового робота.

Исторически первым браузером в современном понимании (т.е. с графическим интерфейсом и т.д.) была программа NCSA Mosaic, разработанная Марком Андерисеном и Эриком Бина. Mosaic имел довольно ограниченные возможности, но его открытый исходный код стал основой для последующих разработок.

Ныне браузер — комплексное приложение для обработки и вывода разных компонентов веб-страницы.

Наибольшую распространенность имеют следующие браузеры: Chrome (Google), Internet Explorer (Microsoft), Firefox (Mozilla Foundation), Safari (Apple), Opera (Opera Software ASA) и браузеры различные мобильных устройств (Рисунок 2).

Google Chrome — браузер с открытым исходным кодом, разрабатываемый компанией Google. Первая стабильная версия вышла 11 декабря 2008 года. В отличие от многих других браузеров, в Chrome каждая вкладка является отдельным процессом. В случае если процесс обработки содержимого вкладки зависнет, его можно будет завершить без риска потери данных других вкладок. Еще одна особенность - интеллектуальная адресная строка (Omnibox). К возможности автозаполнения она добавляет поисковые функции с учетом популярности сайта, релевантности и пользовательских предпочтений (истории переходов). Планируется кроссплатформенность.

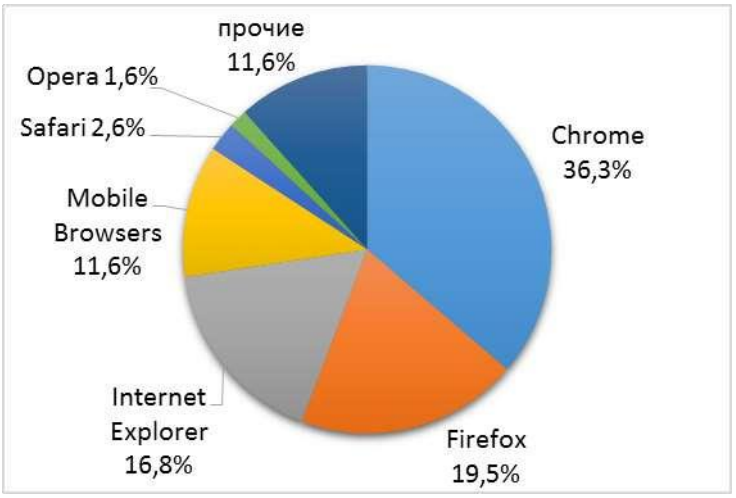


Рисунок 2 - Статистика использования браузеров за период 04.2014 г - 04.2015 г. (по данным statcounter.com)

Internet Explorer (IE) - браузер, разработанный компанией Майкрософт, является наиболее распространенным в мире. Это связано с тем, что IE тесно интегрировансОС Windowsи поставляется вместе с ней. Платформозависим (поддержка сторонних ОС прекращена, начиная с версии 5). Единственный браузер, напрямую поддерживающий технологию ActiveX. Не полностью совместим со стандартами W3C, в связи с чем требует дополнительных затрат от веб-разработчиков.

Firefox - свободный кроссплатформенный браузер, разрабатываемый Mozilla Foundation и распространяемый под тройной лицензией GPL/LGPL/MPL.В основе браузера - движок Gekko, который изначально создавался для Netscape Communicator. Однако, вместо того, чтобы предоставить все возможности движка в стандартной поставке, Firefox реализует лишь основную его функциональность, предоставляя пользователям возможность модифицировать браузер в соответствии с их требованиями через поддержку расширений (add-ons), тем оформления и плагинов.

Safari - проприетарный браузер, разработанный корпорацией Apple и входящий в состав операционной системы Mac OSX. Бесплатно распространяется для операционных систем семейства Microsoft Windows. В браузере используется уникальный по производительности интерпретатор JavaScript, опережающий аналоги более чем в 2 раза. Среди других особенностей: функция «Snapback», которая позволяет мгновенно вернуться к исходным результатам поиска или к верхнему уровню любого веб-сайта; те же графические технологии, что и в Mac OS X; предварительная поддержка стандартов CSS3 и HTML5; автоматическое распознавание и подгрузка отсутствующих в системе шрифтов; интеграция мультимедийных технологий QuickTime.

Opera — кроссплатформенный многофункциональный веб-браузер, разработанный в 1994 году группой исследователей из норвежской компании Telenor. Дальнейшая разработка ведется Opera Software ASA. Браузер Operaобладает высокой скоростью работы и совместим с основными

стандартами WWW. Отличительными особенностями Operадолгое время являлись многостраничный интерфейс (система вкладок) и возможность масштабирования веб-страниц целиком, вместе с графикой. Функция Small Screen Renderingпозволяет переформатировать документ для отображения на экранах мобильных устройств. В Opera, кроме стандартных способов навигации предусмотрены так называемые «жесты мышью». На разных этапах развития в Operабыли интегрированы возможности почтового/новостного клиента Opera Mail, адресной книги, клиента сети BitTorrent, агрегатора RSS, клиента IRC, менеджера закачек, WAP-браузера, а также поддержка виджетов — графические модулей, работающих вне окна браузера.

1.3 Язык разметки гипертекстовых документов HTML

Язык разметки документов - это набор специальных инструкций, называемых тэгами, предназначенных для формирования в документах какой-либо структуры и определения отношений между различными элементами этой структуры. Тэги языка, или, как их иногда называют, управляющие дескрипторы, в таких документах каким-то образом кодируются, выделяются относительно основного содержимого документа и служат в качестве инструкций для программы, производящей показ содержимого документа на стороне клиента. В самых первых системах для обозначения этих команд использовались символы < и >, внутри которых помещались названия инструкций и их атрибуты. Сейчас такой способ обозначения тэгов является стандартным.

Самый популярный на сегодняшний день язык гипертекстовой разметки HTML, был создан специально для организации информации, распределенной в сети Интернет, и является одной из ключевых составляющих технологии WWW.

HTML (Hyper Text Markup Language) - Язык гипертекстовой разметки, который в настоящее время используется в World Wide Web. Изначально создавался как язык для обмена научной и технической документацией. Стандартизацией языка HTML занимается W3C (WWW Consortium).

HTML является упрощенной версией стандартного общего языка разметки - SGML (Standart Generalised Markup Language), который был утвержден ISO в качестве стандарта еще в 80-х годах.

В качестве основы написания кода HTML был выбран обычный текстовый файл. Таким образом, гипертекстовая база данных в концепции WWW — это набор текстовых файлов, размеченных на языке HTML, который определяет форму представления информации (разметка) и структуру связей между этими файлами и другими информационными ресурсами (гипертекстовые ссылки).

Разработчики HTML смогли решить две задачи:

- предоставить дизайнерам гипертекстовых баз данных простое средство создания документов;

- сделать это средство достаточно мощным, чтобы отразить имевшиеся на тот момент представления об интерфейсе пользователя гипертекстовых баз данных.

Большинство документов имеют стандартные элементы, такие, как заголовки, параграфы или списки. Используя тэги (команды) HTML, можно обозначать данные элементы, обеспечивая браузеры минимальной информацией для их отображения, сохраняя в целом общую структуру и информационную полноту докумен-тов. В большинстве случаев автор документа строго определяет внешний вид документа. В случае HTML читатель (основываясь на возможностях браузера) может в определенной степени управлять внешним видом документа (но не его содержимым). HTML позволяет отметить, где в документе должен быть заголовок или абзац, при помощи тэга HTML, а затем предоставляет браузеру интерпретировать эти тэги.

Общая структура тэга и его содержимого такова: <тэг атрибут_1=значение_1 атрибут_2=значение_2 ... атрибут_К=значение_К> содержимое элемента</тэг>. Одним из принципов языка HTML является многоуровневое вложение элементов.

Любой HTML-документ имеет следующую структуру:

```
<html>
<head>
<!-- заголовок документа -->
</head>
<body>
<!-- содержание документа -->
</body>
</html>
```

Тег <html> является самым внешним, так как между его стартовым и конечным тегами должна находиться вся Web-страница. Конечным тегом </html> заканчиваются все гипертекстовые документы.

Заголовок содержит служебную информацию, в частности, предназначенную для поисковых систем.

Все тэги, которые предназначены для оформления документа, могут быть условно разделены на несколько групп:

- форматирование;
- верстка таблиц;
- верстка списков;
- формирование гиперссылок;
- вставка изображений.

Тэт верстки, таблиц позволяют формировать и отображать таблицы произвольной сложности. Вообще дизайнеры довольно часто используют таблицы для оформления страниц, помещая в них меню, текст, рисунки и т.д.

Тэги верстки списков позволяют формировать маркированные и нумерованные списки.

Гипертекстовый документ невозможно представить себе без ссылок на другие документы (внутренние или внешние). Ссылки формирует тэг <A>...</A> - с обязательным атрибутом HREF.

Тэг для отображения рисунков — <IMG>. Он не имеет закрывающегося тэга и содержит обязательный атрибут SRC, значением которого является адрес фай-ла с рисунком {относительный, т.е. на данном сайте, но, например, в другом каталоге, или абсолютный, если рисунок, например, изображение счетчика, подгружает-ся с другого сайта).

HTML обладает несложным набором команд и вполне успешно справляется с задачей описания текстовой информации и отображением ее на экране программы просмотра-браузера. Однако сами отображаемые данные никак не связаны с теми тэгами, которые используются для форматирования, поэтому у программ-анализаторов нет возможности использовать тэги HTML для поиска нужных нам фрагментов документа.

Другим существенным недостатком HTML можно назвать ограниченность набора его тэгов. DTD - правила для HTML определяют фиксированный набор дескрипторов и поэтому у разработчика нет возможности вводить собственные, специальные тэги. Хотя время от времени появляются новые расширения языка, но долгий путь их стандартизации, сопровождаемый постоянными разногласиями между основными производителями браузеров делают практически невозможной быструю адаптацию языка, его использование для отображения специализированной информации (например, мультимедийной, математических, химических формул и т.д.).

Различают два вида html-документов – статические и динамические. Статические документы хранятся в файлах той файловой системы, которая используется web-сервером или браузером при просмотре локальных файлов. При размещении информации на web-сервере можно использовать динамические документы - такие, которые не существуют постоянно в виде файлов, а генерируются в момент запроса клиента. При чем для конечного пользователя не имеет значения динамический или статический способ представления документов.

Для генерирования динамического документа HTML требуется специально написанная программа по правилам, определяемым web-сервером. При планировании размещения информации на web-сервере, для правильного определения использования, какого-либо вида документов, необходимо учитывать степень обновляемости данных, их объем и частоту обращения.

Динамический способ определяет хранение данных в формализованном виде, например, в базе данных.

Если же данные хранятся в формализованном виде, то, используя шаблоны документов, в которых были произведены изменения, генерируются статические документы. Для генерирования статических документов можно использовать любые средства отчетов, имеющихся в той системе управления баз данных (СУБД), которой обработаны и формализованы данные.

Каскадные (многоуровневые) таблицы стилей - cascading style sheets (CSS) - это мощный стандарт на основе текстового формата, определяющий представление данных в браузере.

Если формат HTML предоставляет информацию о составе документа, то таблицы стилей сообщают как он должен выглядеть. Таким образом каскадные таблицы стилей дают возможность хранить содержимое отдельно от его представления.

Стиль включает все типы элементов дизайна: шрифт, фон, текст, цвета ссылок, поля и расположение объектов на странице.

CSS разрабатывались так, чтобы обеспечить больший уровень контроля над размещением текста и графики.

Каскадные таблицы стилей обеспечивают должный уровень единства оформления, организации и контроля во время разработки узла, который является недостижимым с помощью одного только HTML.

1.4 История развития HTML

Свою историю HTML начинает с 1986 года, когда Международной организацией по стандартизации (ISO) был принят стандарт, озаглавленный "Standard Generalized Markup Language" - SGML. Этот стандарт был посвящен описанию обобщенного мета языка, который позволял строить системы логической структурной разметки любых разновидностей текстов. Он соответствовал международному стандарту ISO 8879.

Создатели SGML стремились к тому, чтобы размеченный текст могла без труда интерпретировать любая программа, работающая на разных компьютерных платформах и устройствах вывода.

SGML является не готовой системой разметки текста, а определяет лишь синтаксис записи элементов разметки – тегов и их атрибутов, а также правила определения новых тегов и указания структурных отношений между ними.

Идеология SGML повлияла на многие компьютерные разработки, однако сам по себе язык не получил обширного распространения.

В 1991 году, британским ученым Тимом Бернерсом Ли, сотрудником Европейского института физики частиц (CERN) в Женеве, была разработана система передачи гипертекстовой информации через интернет. А за основу нового языка был взят SGML. Язык разметки гипертекста был назван - HTML (Hyper Text Markup Language) и он является до сих пор самым известным из приложений SGML. HTML в первую очередь был разработан для обмена научной и технической документации для использования людьми, не являющимися специалистами в области верстки. Путем определения небольшого набора структурных и семантических элементов, получались достаточно простые и в тоже время красиво оформленные документы. HTML успешно справился с проблемами SGML.

Как и положено изначально язык HTML разделял все особенности идеологии SGML. Но в 1993 году появляется версия языка HTML 1.2, которая

имела сорок с небольшим тегов, три из которых не рекомендованы к использованию, так как указывали на физические параметры предоставления документа, что противоречило идеологии SGML. Вся разметка была логической и только в описательной части стандарта можно было увидеть что-то типа «в графических браузерах действие этого тега может передаваться жирным начертанием».

Программа Mosaic была единственной в то время браузером, поддерживающим графические возможности. Она была разработана в Национальном центре суперкомпьютерных приложений США (National Center for Supercomputer Applications - NCSA), там же, кстати, была разработана WWW (World Wide Web). По этой причине никаких противоречий между официальными стандартами и их реализацией в браузерах тогда не существовало.

В апреле 1994 года был образован Консорциум W3C (World Wide Web Consortium). Так как официальной спецификации HTML 1.0 не существовало, W3C начал заниматься подготовкой спецификации HTML следующей версии. Но чтобы стандартная версия отличалась от всех предыдущих, ей сразу присвоили номер 2.0. Разработка спецификации HTML 2.0 шла не спеша и лишь в сентябре 1995 года она была утверждена. Из больших дополнений был добавлен лишь механизм форм для отсылки информации с компьютера пользователя на сервер.

Тем временем Консорциум W3 в параллель со спецификацией 2.0 занимался обсуждением HTML 3.0. Она была предложена в марте 1995 года. Третья версия предлагала много новых возможностей: поддержка таблиц, обтекание изображения текстом, отображение сложных математических формул, примечания. Поддержка этого стандарта браузерами того времени была не удовлетворена.

Авторы HTML 3 добавили поддержку нового средства - иерархические стилевые спецификации (Cascading Style Sheets, CSS). Это нововведение нужно было для того, чтобы разрешить назревшее к тому времени противоречие между идеологией структурной разметки и потребностями пользователей, которым в первую очередь нужно было гибкость и обширные возможности визуального представления. CSS имеет свой синтаксис и является формально независимой от HTML.

Следующей версией HTML стала 3.2 и в ней были опущены многие нововведения из версии 3.0.

18 декабря 1997 года бала принята четвертая версия HTML. Она содержала как и третья много элементов, специфичных для отдельных браузеров. Хотя в HTML 4.0 произошла чистка элементов из предыдущих версий спецификаций. Многие элементы были помечены как устаревшие и не рекомендуемые к использованию. Вместо них рекомендовалось использовать таблицы стилей CSS.

HTML 4.0 поддерживал новые мультимедийные возможности, скрипты, таблицы стилей, улучшенную печать и более доступные людям с физическими

недостатками документы. В версии HTML 4.0 также успешно реализованы интернационализация документов, целью которой является сделать Паутину действительно всемирной.

HTML 4.01 утвердили 24 декабря 1999. Изменения, принятые в нем были более значительные, чем, кажется на первый взгляд.

Сейчас W3C занимается разработкой пятой версии языка HTML. Ее черновой вариант стал доступен 20 ноября 2007 года.

В параллель ведется также разработка XHTML (Extensible Hypertext Markup Language) - расширяемый язык разметки гипертекста. В нем предъявляются более строгие требования к синтаксису, чем в HTML. XHTML 1.0 был утвержден 26 января 2000 года в качестве рекомендации W3C. Вариант XHTML 1.1 одобрен в качестве рекомендации консорциума 31 мая 2001года.

История HTML5 начинается в 2004 году, когда большая часть видных деятелей веб-индустрии, а также крупных компаний таких как (Google, Mozilla, Opera, Apple и Microsoft), создают свою собственную рабочую группу под названием WHATWG (возглавил её гениальный программист - Ян Хиксон).

Перед тем как начать работать над HTML5, рабочая группа WHATWG создала две спецификации: Web Forms 2.0 (веб-формы) и Web Apps 1.0 (веб-приложения). Затем эти две спецификации объединили и сделали частью спецификации HTML5.

К 2012-му году уже практически все современные браузеры в мире понимают язык HTML5, хотя еще остаются некоторые теги которые браузеры пока не понимают.

HTML5 — это не продолжатель языка разметки гипертекста, а новая открытая платформа, предназначенная для создания веб-приложений использующих аудио, видео, графику, анимацию и многое другое.

С момента релиза HTML5 одним из главных преимуществ нового языка считается его способность работать на мобильных устройствах.

Применение в мобильной среде является очевидным, но не единственным преимуществом HTML5. Сама структура нового стандарта позволяет поисковым системам заглядывать внутрь созданных с его помощью сайтов и приложений, чего нельзя сказать о Flash.

2 Лабораторные работы

Введение в язык HTML и CSS

Язык разметки гипертекста HTML (Hypertext Markup Language) используют для создания Web-документов. Гипертекст, то есть расширенный текст, включает дополнительные элементы: иллюстрации, ссылки, вставные объекты. Под разметкой понимается использование специальных кодов, легко отделяемых от смыслового содержания документа и используемых для реализации гипертекста. Применение этих кодов подчиняется строгим правилам, определяемым спецификацией языка HTML.

HTML-документ - это файл, содержащий обыкновенный текст со специальными командами. Такой файл имеет расширение **.htm** или **.html** и может быть подготовлен в произвольном текстовом редакторе (существуют, однако, специальные программы-конверторы и HTML-редакторы). Создание Web-страниц непосредственно в HTML-кодах надежнее всего в простых текстовых редакторах, например, в приложении Блокнот. В качестве альтернативы можно использовать более функциональный редактор Notepad++, который поддерживает подсветку синтаксиса большого количества языков программирования и разметки, в том числе и HTML.

HTML-документ состоит из содержимого, то есть собственно полезной информации, и команд, задающих структуру. Каждая команда (управляющая конструкция) HTML-документа (тег) должна заключаться в угловые скобки - вот так: <тег>.

Выделяют 2 группы тегов: одиночные, не имеющие закрывающего тега и парные теги (теги-контейнеры), требующие закрывающего тега.

Чаще всего в документе встречаются парные теги (открывающий и соответствующий ему закрывающий), так как браузеру необходимо знать область действия тега. Одиночные теги используются только там, где область действия очевидна и дополнительной информации не требуется (ясно, например, что если мы встретили тег "начало абзаца" (**<P>**), то предыдущий абзац уже закончился). В сомнительном же случае лучше перестраховаться и поставить закрывающий парный тег, иначе документ может оказаться нечитаемым. Открывающий и закрывающий теги называются одинаково и отличаются друг от друга только символом "наклонная черта" или "слэш" - "/", который ставится сразу после открывающей угловой скобки закрывающего тега. Закрытие парных тегов выполняется так, чтобы соблюдались правила вложения.

```
<b><i>На этот текст воздействуют два тега</i></b>
```

Кроме того, тег может включать атрибут, дающий дополнительную информацию браузеру. Например, при помощи атрибута можно попросить браузер изменить величину шрифта, ориентацию изображения по отношению к строке следующего за ним текста, поменять цвет фона документа и т.д. В парных тегах атрибуты добавляются только к открывающему тегу. Атрибуты представляют собой дополнительные ключевые слова, отделяемые от ключевого слова, определяющего тег, и от других атрибутов пробелами и размещаемые до завершающего тег символа ">". Способ применения некоторых атрибутов требует указания значения атрибута. Значение атрибута отделяется от ключевого слова атрибута символом "=" (знак равенства) и заключается в кавычки.

```
<h1 align="left">
```

Язык HTMLв большинстве случаев совершено равнодушен к регистру, в котором набираются теги. Скажем, браузеру совершенно все равно, наберете вы тег, служащий для рисования горизонтальной линии, как **<HR>** или **<hr>** - эффект будет один и тот же.

HTML не признает никакого дополнительного форматирования текста, кроме как с помощью тегов. В результате текст, превосходно смотрящийся в текстовом редакторе, в окне браузера сольется в единую нечитаемую массу. Так, на месте нескольких пробелов будет лишь один пробел. Исчезнут все заголовки, пустые строки, деление текста на абзацы. Без HTML-тегов браузер просто игнорирует все элементы форматирования.

Определение HTML как языка разметки основывается на том, что при удалении из документа всех тегов получается текстовый документ, совершенно эквивалентный по содержанию исходному гипертекстовому документу. Таким образом, при отображении документа HTML сами теги не отображаются, но влияют на способ отображения остальной части документа.

Задание №1. Создание простейшего HTML-документа. Форматирование текста

Рассмотрим последовательность создания и редактирования простейшего HTML-документа.

Задание

Откройте текстовый редактор Блокнот (меню Пуск→Все программы→Стандартные).

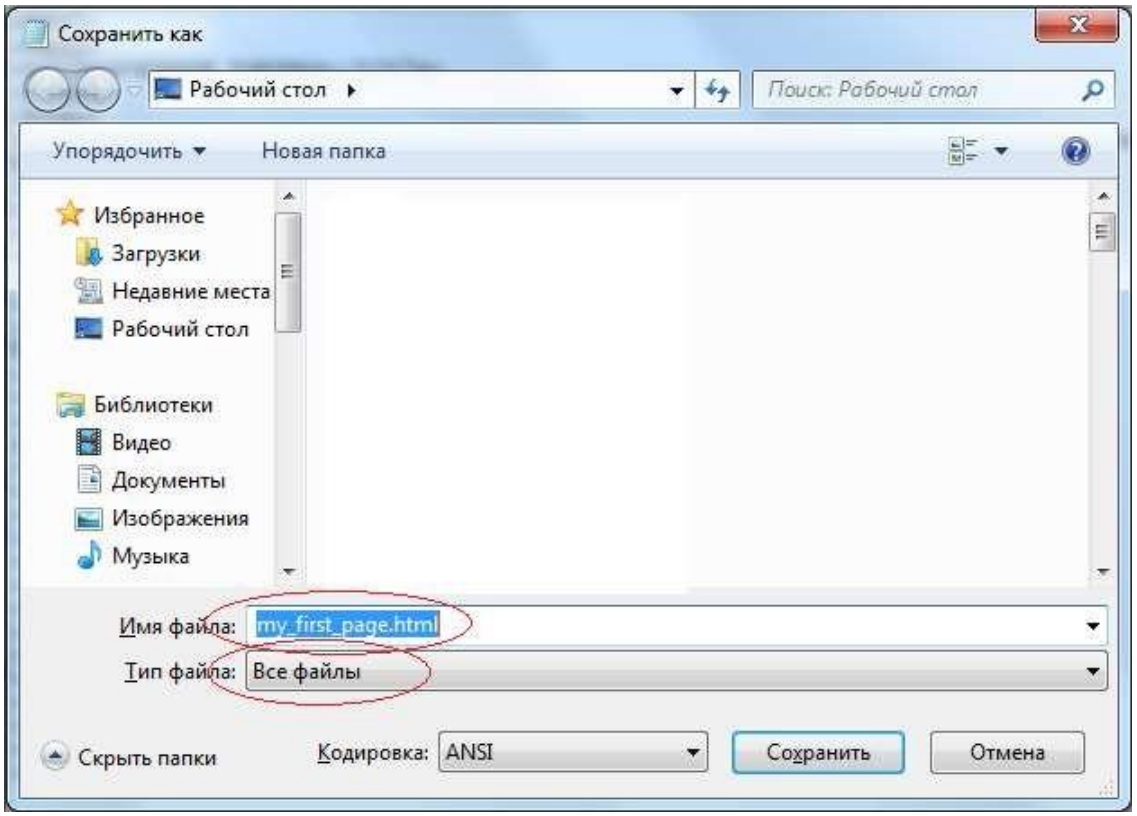
1. В окне текстового редактора введите следующий текст:

```
<html>
<head>
<title>Моя первая страница</title>
</head>
<body>
<p>Привет!
</body>
</html>
```

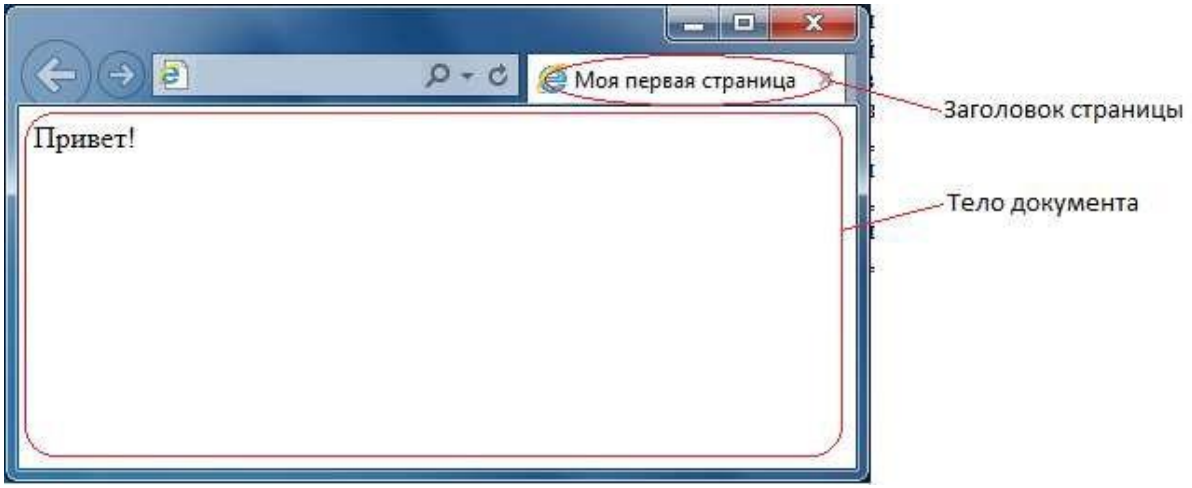
Это пример простейшей структуры html-документа.

Первый тег в документе - **<HTML>**, сообщает браузеру о том, что он имеет дело именно с документом в формате HTML. Тег **<HTML>** и парный ему закрывающий тег **</HTML>** можно считать, как бы "конвертом", в который помещается весь документ. Любой HTML-документ состоит из заголовка, который задается при помощи тега **<TITLE>**, и тела документа, который определяется тегом **<BODY>**. В заголовке документа размещается служебная информация, комментарии автора и заголовок страницы, заключаемый в теги **<TITLE>**. Заголовок, вписанный между тегами **<TITLE>**, в основное текстовое поле браузеру не попадает, а, как правило, размещается в заголовке окна браузера. Тег **<P>** обозначает абзац в тексте.

2. Сохраните документ, используя пункт "Сохранить как…" в меню "Файл". Укажите тип файла – "Все файлы" и дайте ему название, например, my_first_page.html. Обратите внимание, что имя файла обязательно должно заканчиваться расширением .html или .htm.



3. Откройте полученный HTML-документ в любом интернет-браузере, например, Internet Explorer. Документ при этом будет иметь приблизительно следующий вид:



4. Не закрывая браузер, отредактируйте документ в Блокноте, внеся в нем следующие изменения:

- поменяйте заголовок страницы:

```
<title>Страница приветствия</title>
```

- после строки

```
<p>Привет!
```

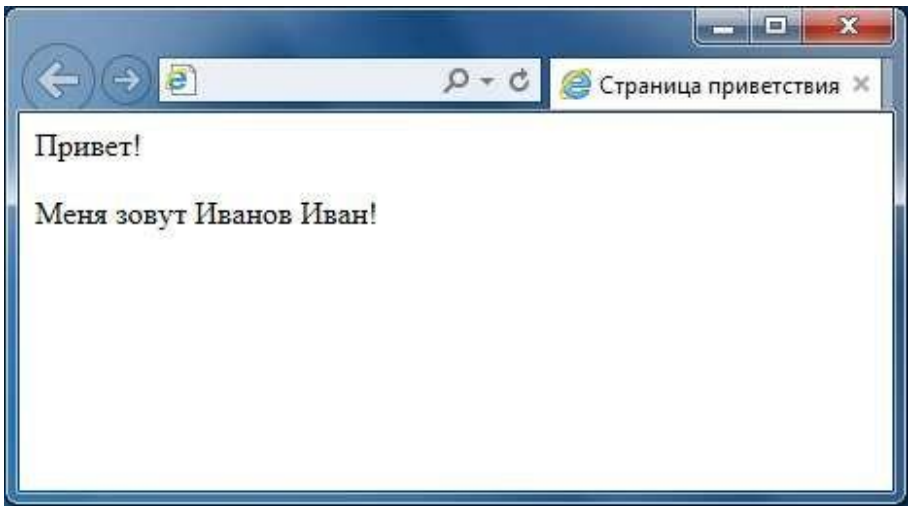
добавьте строку:

```
<p>Меня зовут Иванов Иван!
```

при заполнении используйте свои имя и фамилию.

5. Сохраните документ (Файл→Сохранить).

6. Для отображения в браузере изменений документа, нажмите кнопку F5 на клавиатуре.



При необходимости можно просмотреть HTML-код в самом браузере.

Большинство элементов языка HTML описывает части содержания документа и помещается между тегами **<BODY>** и **</BODY>**, то есть, внутрь структурного элемента BODY. Такие элементы делят на блочные и текстовые. Блочные элементы относятся к частям текста уровня абзаца. Текстовые элементы описывают свойства отдельных фраз и еще более мелких частей текста. Теперь можно сформулировать правила вложения элементов:

• элементы не должны пересекаться (т.е. если открывающий тег располагается внутри элемента, то и соответствующий закрывающий тег должен располагаться внутри этого же элемента);

• блочные элементы могут содержать вложенные блочные и текстовые элементы;

• текстовые элементы могут содержать вложенные текстовые элементы;

• текстовые элементы не могут содержать вложенные блочные элементы.

Строго говоря, все правила языка HTML можно рассматривать исключительно как "пожелания". Средство, используемое для отображения Web-документа, сделает все возможное, чтобы истолковать разметку наиболее разумным образом. Тем не менее, гарантию правильного воспроизведения документа дает только неукоснительное следование требованиям спецификации языка.

Функциональные блочные элементы

В большинстве документов основными функциональными элементами являются заголовки и абзацы. Язык HTML поддерживает шесть уровней заголовков. Они задаются при помощи парных тегов от **<H1>** до **<H6>**. При отображении Web-документа на экране компьютера эти элементы показываются при помощи шрифтов разного размера.

Обычные абзацы задаются с помощью парного тега **<P>**. Язык HTML не содержит средств для создания абзацного отступа ("красной строки"), поэтому

при отображении на экране компьютера абзацы разделяются пустой строкой. Закрывающий тег **</P>** рассматривается как необязательный. Подразумевается, что он стоит перед тегом, который задает начало очередного абзаца документа. Например:

```
<h1>Заголовок</h1>
<p>Первый абзац
<p>Второй абзац
<h2>Заголовок второго уровня</h2>
```

Заголовок

Первый абзац

Второй абзац

Заголовок второго уровня

Следствием наличия специального тега, определяющего абзац, является тот факт, что обычного символа конца строки, вводимого по нажатию клавиши ENTER, для создания абзацного отступа недостаточно. Язык HTML рассматривает символы конца строки и пробелы особым образом. Любая последовательность, состоящая только из пробелов и символов конца строки, при отображении документа рассматривается как одиночный пробел. Это, в частности, означает, что символ конца строки даже не осуществляет перехода на новую строку (для этой цели используется текстовый элемент, задаваемый непарным тегом **
**).

Текстовое оформление страниц

Для изменения параметров шрифта можно использовать тег **<font>**. Для тега используются следующие атрибуты: **face, size** и **color**.

Атрибут **Face** служит для задания гарнитуры шрифтов, использующихся для текста. Названий шрифтов можно указать несколько, через запятую. В этом случае, если первый указанный шрифт не будет найден, будет использоваться следующий по списку.

Пример 1. Использование атрибута face

```
<font face="Arial, Helvetica, sans-serif">Текст будет написан шрифтом Arial.</font>
```

Size задает размер шрифта в условных единицах от 1 до 7. Средний размер, используемый по умолчанию принят 3. Размер шрифта можно указывать как абсолютной величиной (например, size=4), так и относительной

(например, size=+1, size=-1). В последнем случае размер изменяется относительно базового.

Пример 2. Задание размера шрифта

```
<font size=1>Шрифт размера 1</font><br>
<font size=2>Шрифт размера 2</font><br>
<font size=3>Шрифт размера 3</font><br>
<font size=4>Шрифт размера 4</font><br>
<font size=5>Шрифт размера 5</font><br>
<font size=6>Шрифт размера 6</font><br>
<font size=7>Шрифт размера 7</font><br>
```

Шрифт размера 1
Шрифт размера 2
Шрифт размера 3
Шрифт размера 4
Шрифт размера 5
Шрифт размера 6
Шрифт размера 7

Color определяет цвет текста, который можно задавать с помощью названий цветов или цифровыми значениями в шестнадцатеричном формате. Существует 16 основных цветов, представленных в таблице.

Цвет	Символическое название	Шестнадцатеричное значение
Морской волны	Aqua	#00FFFF
Черный	Black	#000000
Голубой	Blue	#0000FF
Малиновый	Fuchsia	#FF00FF
Серый	Gray	#808080
Зеленый	Green	#008000
Ярко-зеленый	Lime	#00FF00
Темно-красный	Maroon	#800000
Темно-синий	Navy	#000080
Оливковый	Olive	#808000
Пурпурный	Purple	#800080
Красный	Red	#FF0000

	Серебренный	Silver	#C0C0C0
	Темной морской волны	Teal	#008080
	Белый	White	#FFFFFF
	Желтый	Yellow	#FFFF00

Кроме того, на сайте http://www.puzzleweb.ru/html/colors_html.php (и других подобных) имеется расширенная таблица из 140 цветов и удобное средство для их подбора в шестнадцатеричном формате.

Пример 3. Изменение цвета текста
Первая буква этого предложения будет написана шрифтом Arial, красным цветом и увеличенной.

Первая буква этого предложения будет написана шрифтом Arial, красным цветом и увеличенной.

Видоизменение текста - средства его форматирования, такие как выбор начертания шрифта и использование эффектов, позволяющих менять вид текста. В таблице перечислены основные теги, которые применяются для изменения оформления текста.

КодHTML	Описание	Пример
Текст	Жирный текст	**Текст**
Текст	Курсивное начертание текста	*Текст*
<u>Текст</u>	Подчеркнутый текст	Текст
^{Текст}	Верхний индекс	$e=mc^2$
_{Текст}	Нижний индекс	H_2O
<strike>Текст</strike>	Зачеркнутый текст	~~Текст~~
<pre>Т е к с т</pre>	Текст пишется как есть, включая все пробелы	Т е к с т
Текст	Курсивный текст	*Текст*
Текст	Жирный текст	**Текст**

Обычно для создания верхнего или нижнего индекса используется тег **small**, делающий индекс меньше по размеру основного шрифта.

Пример 4. Создание нижнего индекса
Формула серной кислоты:
H_{<small>2</small>}SO_{<small>4</small>}

Формула серной кислоты:
H_2SO_4

Выравнивание текста

Выравнивание текста определяет его внешний вид и ориентацию краев абзаца и может выполняться по левому, правому краю, по центру или по ширине.

КодHTML	Описание	Пример
Текст	Добавляет новый параграф, по умолчанию выровненный по левому краю. Перед параграфом автоматически добавляется пустая строка.	Текст
Текст	Выравнивание по левому краю.	Текст
Текст	Выравнивание по правому краю.	Текст
Текст	Выравнивание по центру.	Текст
Текст	Выравнивание по ширине.	Текст по ширине

Задание 2

1. Поместите на HTML-страницу следующий код:

```
<html>
<body>
<font size="3" face="Times, Palatino, serif" color="#FF0000">Это пример № 1</font> <br>
<font size="2" face="Arial, Helvetica, nonserif" color="#008000">Это пример № 1</font> <br>
<font size="5" face="Verdana, Arial, Courier, Times" color="#FF8000">Это пример№ 1</font>
</body>
</html>
```

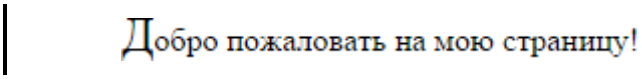
2. Загрузите страницу в браузер и посмотрите, что получилось.

Это пример № 1
Это пример № 1
Это пример № 1

3. Как правило, шрифты типа serif используются для текстов, а шрифты типа nonserif используются для заголовков. Однако, если размер шрифта небольшой тексты serif могут оказаться неразборчивыми. Поэтому старайтесь не использовать минимальные размеры шрифтов. Для упражнения поместите на страницу текст с разными размерами шрифта serif и оцените возможность их легкого прочтения.

4. Иногда для заглавных букв используют размер шрифта, отличный от размера шрифта для остального текста. Поместите на страницу следующую конструкцию и посмотрите в браузере, что получилось:

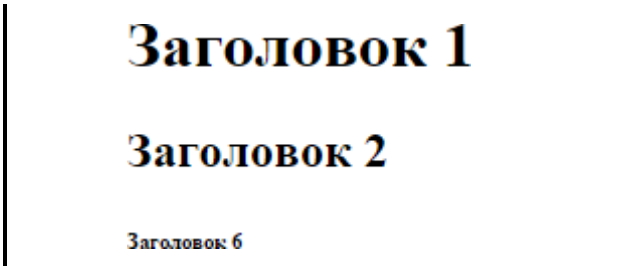
```
<font size="5">Д</font>обро пожаловать на мою страницу!
```



5. Для заголовков часто используют специальные тэги заголовков: <h1>, <h2>, …<h6>.

Используйте следующие примеры для отображения заголовков:

```
<h1>Заголовок 1</h1>
<h2> Заголовок 2</h2>
<h6> Заголовок 6</h6>
```



Изменения фонового оформления страницы

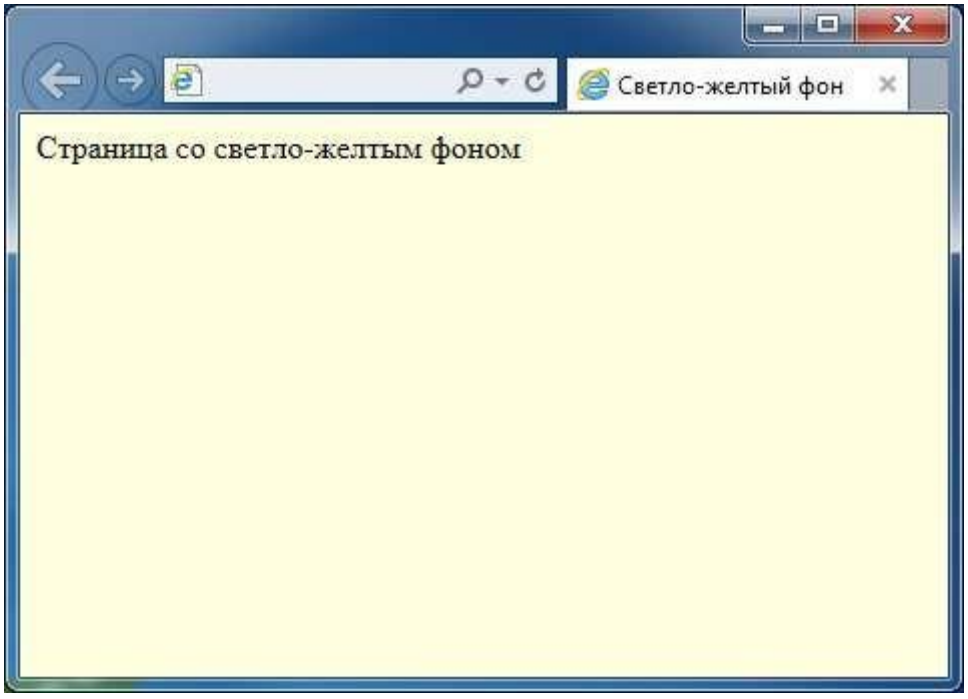
У тега **<BODY>** есть несколько атрибутов, позволяющих поменять фон на странице, цвет текста, а также задать отступы от границ окна.

Цвет фона страницы можно задать с помощью атрибута **bgcolor**.

Пример 5. Страница со светло-желтым фоном

```
<html>
 <head>
   <title>Светло-желтый фон</title>
 </head>
 <body bgcolor="lightyellow">
   Страница со светло-желтым фоном
```

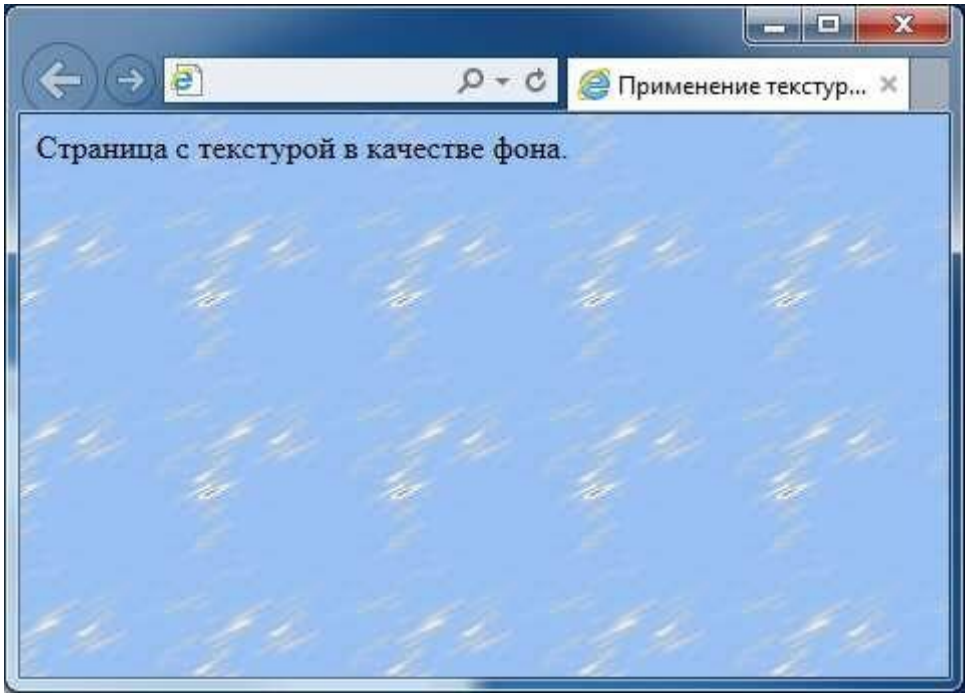
```
 </body>
</html>
```



Атрибут **bgcolor** устанавливает одноцветный фон на странице. Но в качестве фона может быть использована и повторяющаяся картинка, или так называемая текстура. Для этого служит следующий атрибут **background**, позволяющий установить текстурное изображение в качестве фона Web-страницы.

Пример 6. Страница с изображением в качестве текстуры

```
<html>
 <head>
   <title>Применение текстурного фона</title>
 </head>
 <body background="texture.jpg">
   Страница с текстурой в качестве фона.
 </body>
</html>
```



В данном примере в качестве текстуры фона использовалось изображение в файле texture.jpg. Чтобы браузер смог найти изображение-источник, оно должно быть расположено в одной папке с HTML-файлом, для которого устанавливается текстура. В противном случае необходимо прописывать полный адрес изображения источника.

Задания для самостоятельной работы

1. Создайте HTML-документ и оформите в нем текст в соответствии с образцом вашего варианта.
2. С использованием таблицы цветов подберите удобный для восприятия цвет фона страницы.
3. Создайте копию полученного HTML-документа. В качестве фона этого документа используйте текстурное изображение, выбранное вами из папки с текстурами.

Вариант 1

Пусть вечно спорят
дверь с ключом
О ~~первенстве~~ и ~~важности~~.
Главнее нету
в споре том
Глазка *замочной* **скважины**.

Вариант 2

Есть в близости
людей заветная черта,
Ее не перейти влюбленности *и* страсти, —
Пусть в жуткой
тишине сливаются уста,
~~***И сердце рвется от любви на части***~~.

Вариант 3

Все, что в сердце
твоем туманится,
Станет ясно ~~в моей тишине~~,
И когда он с тобой расстанется,
Ты ~~признаешься~~ только **мне.**

Вариант 4

Любовь *глубокой*
~~нежности полна,~~
В соблазнах, горестях закалена,
Крепка в разлуке, вдалеке — горда,
Все та же — чудо, долгие года.

Вариант 5

Жизнь не состязание в борьбе.
С юмором живи,
но без усмешки.
~~Силы пробуждать~~ учись в себе
И живи *обдуманно*, без спешки.

Вариант 6

Октябрь уж наступил,
Уж роща отряхает последние листы
С нагих своих ветвей.
Дохнул осенний хлад,
~~Дорога промерзает,~~
Журча, еще бежит
За мельницу ручей

Вариант 7

И в ночи январской,
беззвездной,
Сам дивясь небывалой *судьбе*,
Возвращенный из смертной **бездны**,
Ленинград ***салютует себе***.

Вариант 8

А тебе еще мало по-русски,
И ты хочешь на всех языках
Знать, ***как круты подъемы*** *и спуски*
~~И почем у нас~~
совесть и страх.

Вариант 9

Что войны, что чума? - конец им виден скорый,
~~Им приговор почти произнесен~~.
Но кто нас защитит от ужаса, ***который***
Был бегом времени
когда-то наречен?

Вариант 10

Д~~орогою ценой~~

и нежданной

Я узнала, что **помнишь** *и* **ждешь**.

А быть может,

и место найдешь

Ты - могилы моей безымянной.

Задание №2. Создание списков

Нумерованные списки

Нумерованные списки представляют собой набор элементов с их порядковыми номерами. Вид и тип нумерации зависит от атрибутов тега **OL**, который и используется для создания списка. В качестве маркеров могут быть следующие значения: арабские цифры заглавные латинские буквы прописные латинские буквы заглавные римские цифры прописные римские цифры

Ниже, в таблице приведены различные атрибуты тега **OL** и результат их применения.

Код HTML	Пример
  - текст - текст - текст 	Нумерованный список с параметрами по умолчанию: 1. текст 2. текст 3. текст
<ol start="5">	Нумерованный список, начинающийся с пяти: 5. текст 6. текст 7. текст
<ol type="A">	Нумерованный список с заглавными буквами латинского алфавита: A. текст B. текст C. текст
<ol type="a">	Нумерованный список с прописными буквами латинского алфавита: a. текст b. текст c. текст

<ol type="I">	Нумерованный список с прописными римскими буквами: I. текст II. текст III. текст
<ol type="i">	Нумерованный список со строчными римскими буквами: i. текст ii. текст iii. текст
<ol type="1">	Нумерованный список с арабскими цифрами: 1. текст 2. текст 3. текст
<ol type-"I" start="7">	Список с римскими цифрами начинающийся с семи: IV. текст V. текст VI. текст

Маркированные списки

Маркированные списки позволяют разбить большой текст на отдельные блоки. Тем самым привлекается внимание читателя к тексту и повышается его читабельность. С учетом худшего восприятия текста с экрана монитора, чем печатного варианта, это является весьма полезным приемом.

Для установки маркированного списка используется тег **UL** и **LI** (Пример 1).

Пример 1. Создание маркированного списка

```
<html>
<body>
Что следует учитывать при тестировании сайта:
<ul>
<li>работоспособность всех ссылок</li>
<li>поддержку разных браузеров</li>
<li>читабельность текста</li>
</ul>
</body>
</html>
```

Ниже показан результат примера 1.

Что следует учитывать при тестировании сайта:

- работоспособность всех ссылок
- поддержку разных браузеров
- читабельность текста

Обратите внимание, что у маркированного текста появляются отступы сверху и снизу. Чтобы от них избавиться, список можно делать без тега **UL**. При этом исчезнут и отступы текста перед маркерами.

Пример 2. Создание маркированного списка без отступов

```
<html>
<body>
Что следует учитывать при тестировании сайта:
<li>работоспособность всех ссылок</li>
<li>поддержку разных браузеров</li>
<li>читабельность текста</li>
</body>
</html>
```

Ниже показан результат примера 2.

Что следует учитывать при тестировании сайта:

- работоспособность всех ссылок
- поддержку разных браузеров
- читабельность текста

Маркеры могут принимать один из трех видов: круг (по умолчанию), окружность и квадрат. Для выбора типа маркера используется атрибут **type="..."** тега **UL**. Вместо многоточия подставляется одно из трех значений, указанных в таблице.

Код **HTML**	**Пример**
<ul type="disc">	Что следует учитывать при тестировании сайта: • работоспособность всех ссылок • поддержку разных браузеров • читабельность текста
<ul type="circle">	Что следует учитывать при тестировании сайта: ○ работоспособность всех ссылок ○ поддержку разных браузеров ○ читабельность текста

<ul type="square">	Что следует учитывать при тестировании сайта: ■ работоспособность всех ссылок ■ поддержку разных браузеров ■ читабельность текста

С помощью CSS этот список можно расширить и вместо встроенных символов использовать в качестве маркера рисунок.

Многоуровневые списки

Многоуровневые списки применяются на веб-страницах для создания сложной структуры текста либо для использования у многоуровневых меню. Создать многоуровневый список HTML совсем не сложно, достаточно в один из элементов списка **<LI> </LI>** вложить тег **<UL>** или **<OL>** с новым элементами списка вот так выглядит код подобной конструкции:

Пример 3. Создание многоуровневого списка

```
<html>
<body>
Основные устройства компьютера:
<ol>
<li>монитор</li>
<li>системный блок
<ul>
<li>материнская плата</li>
<li>процессор</li>
</ul>
</li>
<li>клавиатура</li>
<li>мышь</li>
</ol>
</body>
</html>
```

Результат выглядит следующим образом:

Основные устройства компьютера:

1. монитор
2. системный блок
   - материнская плата
   - процессор
3. клавиатура
4. мышь

Как видно из примера вложенный список получаются, путем вставки одного списка в другой. То есть один список вставляется в элемент «li» другого списка. Единственное неудобство, что такой вид организации списков легко может привести к путанице.

Графические маркеры списка

В качестве маркеров списка можно использовать графические изображения, что широко применяется для создания привлекательных, красиво оформленных HTML-документов. На самом деле такая возможность не предоставляется непосредственно языком HTML, а реализуется несколько искусственно.

Чтобы понять идею, необходимо разобраться в механизме реализации списков на HTML-страницах. Оказывается, что тег списка **<UL>** (как, впрочем, и теги списков других типов, рассматриваемых ниже) выполняет единственную задачу — указывает браузеру, что вся информация, располагаемая после данного тега должна отображаться со сдвигом вправо (отступом) на некоторую величину. Теги **<LI>**, указывающие на отдельные элементы списка, обеспечивают вывод стандартных маркеров элементов списка.

Если же нам требуется построить список с графическими маркерами, то можно вообще обойтись без тегов **<LI>**. Достаточно будет перед каждым элементом списка вставить желаемое графическое изображение. Единственной задачей, которую нужно при этом решить, будет отделение элементов списка друг от друга. Для этого можно использовать теги абзаца **<P>** или принудительного перевода строки **
**.

Пример 4. Создание списка с графическими маркерами

```
<html>
<head>
<title>Маркированный список</title>
</head>
<body>
<ul>
<b>Знаки зодиака:</b><br>
<img src="gold_star.jpg"> Овен<BR>
<img src="gold_star.jpg"> Телец<BR>
<img src="gold_star.jpg"> Близнецы<BR>
<img src="gold_star.jpg"> Рак<BR>
<img src="gold_star.jpg"> Лев<BR>
<img src="gold_star.jpg"> Дева<BR>
<img src="gold_star.jpg"> Весы<BR>
<img src="gold_star.jpg"> Скорпион<BR>
<img src="gold_star.jpg"> Стрелец<BR>
```

```
<img src="gold_star.jpg"> Козерог<BR>
<img src="gold_star.jpg"> Водолей<BR>
<img src="gold_star.jpg"> Рыбы
</ul>
</body>
</html>
```

В окне браузера список выглядит следующим образом:

Знаки зодиака:

- Овен
- Телец
- Близнецы
- Рак
- Лев
- Дева
- Весы
- Скорпион
- Стрелец
- Козерог
- Водолей
- Рыбы

В приведенном примере в качестве маркера элементов списка используется графический файл gold_star.jpg. Заметим, что использование графики на HTML-страницах может значительно увеличить объем передаваемой информации. Однако в данном случае это увеличение крайне незначительно. Здесь для всех маркеров используется один и тот же файл, который будет передан только один раз. Размеры файла, содержащего маленькое изображение, также крайне незначительны.

Задания для самостоятельной работы

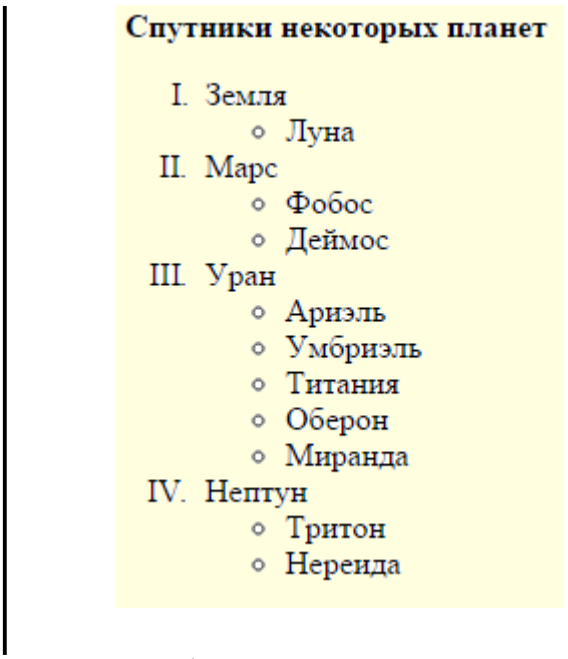
1. Создайте HTML-страницу с двухуровневым списком спутников некоторых планет и оформите согласно образцу используя типы маркера или нумерации, взятые из таблицы.

Таблица – Варианты оформления списков

№ варианта	1-ый уровень	2-ой уровень
1	Арабские цифры	Маркер "square"
2	Римские цифры строчные	Маркер "circle"
3	Римские цифры прописные	Арабские цифры
4	Прописные буквы	Римские цифры строчные

5	Маркер "disc"	Римские цифры прописные
6	Маркер "square"	Арабские цифры
7	Маркер "circle"	Римские цифры строчные
8	Арабские цифры	Римские цифры прописные
9	Римские цифры строчные	Прописные буквы
10	Римские цифры прописные	Маркер "disc"

Образец двухуровневого списка, в котором для первого уровня используются римские прописные цифры, а для второго - маркер "circle":



2. Оформите страницу используя цветной фон или фоновую текстуру.

3. Создайте рисунок небольшого размера в виде какого-либо значка или символа в графическом редакторе MS Paint (или любом другом). Измените созданный в задании 1 двухуровневый список используйте в качестве маркера для первого уровня собственный рисунок.

Задание №3. Создание гиперссылок

Важнейшим свойством языка HTML является возможность размещения на странице ссылок на другие документы. Возможны ссылки:

- на удаленный HTML файл;
- на некоторую точку в текущем HTML-документе;
- на любой файл, не являющийся HTML-документом.

В качестве ссылки можно использовать текст или графику.

Для создания ссылки необходимо сообщить браузеру, что является ссылкой, а также указать адрес документа, на который следует сделать ссылку. Оба действия выполняются с помощью тега **A**, который имеет единственный атрибут **href**. В качестве значения используется адрес документа: для

документа в сети интернет - это его URL, для документа на компьютере пользователя - это имя и путь к нему.

Адрес ссылки может быть абсолютным и относительным. Абсолютные адреса работают везде и всюду независимо от имени сайта или расположения html-страницы, где прописана ссылка.

Пример 1. Использование абсолютных ссылок

```
<html>
<body>
<a href=http:\\www.yandex.ru>Абсолютная ссылка на сайт</a>
<a href="c:\docs\user\main.html">Абсолютная ссылка на другой локальный html-документ</a>
</body>
</html>
```

Относительные ссылки, как следует из их названия, построены относительно текущего документа или адреса. Примеры таких адресов:

1. /
2. /demo/
3. /images/pic.gif
4. ../help/me.html
5. manual/info.html

Первые две ссылки называются неполные и указывают веб-серверу загружать файл index.html (или default.html) находящемуся в корне сайта (пример 1) или папке demo (пример 2). Если файл index.html отсутствует, браузер, как правило, показывает список файлов, находящихся в данном каталоге. Слэш перед адресом говорит о том, что адресация начинается от корня сайта (пример 3), двоеточие - перейти на уровень выше в списке каталогов сайта (пример 4).

Пример 2. Использование относительных ссылок

```
<html>
<body>
<a href=images/photo.jpg>Относительная ссылка на рисунок</a><br>
<a href=main.html> Относительная ссылка на другой html-документ или страницу</a>
</body>
</html>
```

Ссылки внутри страницы

Большие документы читаются лучше, если они имеют оглавление со ссылками на соответствующие разделы. Для создания ссылки следует вначале сделать закладку в соответствующем месте и дать ей имя при помощи атрибута name тега **A.**

Пример 3. Создание внутренней ссылки

```
<html>
<body>
<a name=top></a>Друг уронил утюг в унитаз. И разбил его. Причем так разбил, что по назначению унитаз и использовать никак нельзя.
Мгновением назад только что вот все было хорошо и вот уже дыра, да такая, что можно забыть, что есть такой предмет в доме. Махнул рукой нечаянно, а потом мучайся...
<a href=#top>Наверх</a>
</body>
</html>
```

Между тегами **** и **** отсутствует текст, так как требуется лишь указать местоположение перехода по ссылке, находящейся внизу страницы. Имя ссылки на закладку начинается символом #, после чего идет название закладки. Название выбирается любое, соответствующее тематике.

Можно, также, делать ссылку на закладку, находящуюся в другой веб-странице и даже другом сайте. Для этого в адресе ссылки надлежит указать ее адрес и в конце добавить символ решетки # и имя закладки.

Пример 4. Ссылка на закладку из другой веб-страницы

```
<html>
<body>
<a href=text.html#bottom>Перейти к нижней части текста</a>
</body>
</html>
```

Ссылка на новое окно

Если требуется сделать ссылку на документ, который открывается в новом окне браузера, используется атрибут **target=_blank** тега **A.**

Создание нового окна обычно требуется в случаях, когда делается ссылка на другой сайт, в остальном лучше открывать документы в текущем окне, поскольку обилие окон может сбить читателя с толку.

Так как ссылки на текущее или новое окно ничем не отличаются друг от друга, на некоторых сайтах рядом со ссылкой ставят специальную иконку, показывающую, что документ открывается в новом окне.

Пример 5. Создание ссылки на новое окно

```
<html>
<body>
<a href=http:\\www.igsha.ru>Обычная ссылка на сайт www.igsha.ru</a><br><a href=http:\\www.igsha.ru target=_blank>Ссылка открывает новое окно на сайт www.igsha.ru</a>
</body>
</html>
```

Обычная ссылка на сайт www.igsha.ru
Ссылка открывает новое окно на сайт www.igsha.ru

С помощью HTML можно задавать цвета ссылок на странице.
Для этого используются следующие атрибуты тега **<BODY>**.

1. **link** — определяет цвет ссылок на веб-странице. Цвет по умолчанию синий, #0000FF.
2. **alink** — цвет активной ссылки. Цвет ссылки меняется при нажатии на ней кнопки мыши. Цвет по умолчанию красный, #FF0000.
3. **vlink** — цвет уже посещенных ссылок. Цвет по умолчанию фиолетовый, #800080.

Пример 6. Изменение цвета ссылок

```
<html>
<body link="red" vlink="grey" alink="blue" bgcolor="lightyellow">
 <p><a href="content.html">Содержание сайта</a></p>
 </body>
</html>
```

В данном примере цвет ссылок установлен красным, а цвет уже посещенных ссылок - серым.

Ссылка перед нажатием и после него:

Ссылка на сайт www.igsha.ru Ссылка на сайт www.igsha.ru

Задания для самостоятельной работы

1. Создайте в html-документе словарь терминов, который содержит внутренние ссылки для перехода по заглавным буквам терминов и на начало документа согласно образцу. Тематику словаря выбрать из таблицы 1.

Таблица 1 – Варианты тематик словаря терминов

№ варианта	Тематика словаря терминов
1	Информатика
2	Информационные технологии
3	Физика
4	География
5	Биология
6	Экономика
7	Управление
8	Право
9	Социология
10	Общество

Образец (фрагмент словаря терминов)

Информатика - словарь терминов

А Б В Г Д Е Ж З И К

А

АРХИТЕКТУРА ФОН НЕЙМАНА
архитектура компьютера, имеющего одно арифметико-логическое устройство, через которое проходит поток данных, и одно устройство управления, через которое проходит поток команд.

АСИНХРОННАЯ ПЕРЕДАЧА ДАННЫХ
способ передачи и метод извлечения данных из непрерывного потока сообщений, при которых передающая сторона в каждое данное вводит стартовый и стоповый биты, указывающие, где данное начинается и где кончается.

В начало

Б

БАЙТ
машинное слово минимальной размерности, адресуемое в процессе обработки данных. Размерность байта - 8 бит - принята не только для представления данных в большинстве компьютеров, но и в качестве стандарта для хранения данных на внешних носителях, для передачи данных по каналам связи, для представления текстовой информации.

БОД
единица измерения скорости передачи данных.

В начало

В

В качестве источника информации для словаря терминов можно использовать сайт http://www.glossary.ru или другие. Словарь должен содержать не менее 6 групп терминов с одинаковыми заглавными буквами (начиная от буквы А) по 4-5 термина в каждом (если имеются).

2. Оформите html-документ используя цветной фон и разные типы шрифтов для названий терминов и их определений. Добавьте горизонтальные разделительные линии между группами терминов по заглавным буквам используя одиночный тег **<HR>**. Сохраните полученный документ под именем glossary.html.

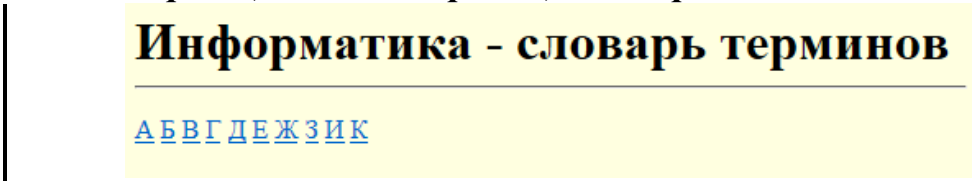
3. Создайте словарь терминов состоящий из нескольких html-документов: главной страницы и страниц групп терминов, переход между которыми осуществляется с использованием внешних ссылок. Для этого выполните следующее.

3.1. Для каждой группы терминов создайте собственные html-документы, например, group_a.html, group_b.html и т.д.

Образец (содержание group_a.html - группа терминов на букву А)

3.2. Создайте документ main.html содержащий только название словаря терминов и внешние относительные ссылки на html-документы групп терминов (group_a.html, group_b.html и т.д.).

Образец главной страницы словаря



Для переходов между страницами используйте относительные ссылки.

Задание №4. Вставка изображений в html-страницы

Для вставки изображений в HTML применяются два основных формата GIF и JPEG. Формат GIF может хранить внутри себя простейшую анимацию (динамические баннеры), JPEG отлично подходит для изображений с большим количеством цветов, например, фотографий. Третьим форматом для web-графики является формат PNG, но он не получил широкого применения в web-дизайне. Любое изображение в форматах GIF или JPEG вставляется на web-страницу при помощи тега **<IMG>**, закрывающего тега нет.

Посредством атрибута **SRC** задается адрес (URL) файла с изображением, т.е. браузер находит нужное изображение в каталоге сайта по пути (адресу URL), прописанному в этом атрибуте. Для указания адреса изображения можно задавать как абсолютный, так и относительный адрес. Для удобства все изображения сайта находятся в отдельной папке, обычно с именем **image**.

Для примера возьмите любое изображение, лучше небольшого формата, и сохраните в созданной папке image. Пусть это будет эмблема нашего вуза с именем **irgau.jpg**. Далее мы будем обращаться к ней для обучения.

Пример 1. Вставка изображения

```
<html>
<body>
<img src="image/irgau.jpg">
</body>
</html>
```

Браузер отобразит:

Теперь попробуем вставить изображение с текстом:

Пример 2. Вставка изображения с текстом

```
<html>
<body>
<p>Текст в который просто вставлена
картинка.<img src="image/irgau.jpg">Обтекание не задано.</p>
</body>
</html>
```

Текст в который просто вставлена картинка.  Обтекание не задано.

По умолчанию, если не задано никаких параметров обтекания, рисунок располагается на странице как объект строки (слово, буква и т.д.). Высота строки при этом увеличивается до размеров рисунка, а текст не распределяется вокруг изображения. Обтекание задается с выравниванием изображения относительно правого или левого края страницы. После чего рисунок вставляется на нужное место, а текст располагается с другой стороны.

Выравнивание изображений

Для изображений можно указывать их положение относительно текста или других изображений на веб-странице. Способ выравнивания изображений задается атрибутом **align** тега **IMG**. В таблице перечислены возможные значение этого атрибута и результат его использования.

Код HTML	Описание	Пример
<img src="smile.jpg" align=**texttop**>	Верхняя граница изображения выравнивается по самому высокому текстовому элементу текущей строки.	Lorem ipsum dolor sit amet,  consectetuer adipiscing elit...
<img src="smile.jpg" align=**top**>	Верхняя граница изображения выравнивается по самому высокому элементу текущей строки.	Lorem ipsum  dolor sit amet, consec-tetuer adipiscing elit...
<img src="smile.jpg" align=**middle**>	Выравнивание середины изображения по базовой линии текущей строки.	Lorem ipsum dolor sit amet,  consec-tetuer adipiscing elit...
<img src="smile.jpg" align=**absmiddle**>	Выравнивание середины изображения посередине текущей строки.	Lorem ipsum dolor sit amet,  consec-tetuer adipiscing elit...
<img src="smile.jpg" align=**baseline**>	Выравнивание изображения по базовой линии текущей строки.	Lorem ipsum dolor sit amet, consectetuer  adipiscing elit...

<img src="smile.jpg" align=**bottom**>	Выравнивание нижней границы изображения по окружающему тексту.	Lorem ipsum dolor sit amet, consectetuer adipiscing elit. ..
<img src="smile.jpg" align=**left**>	Выравнивает изображение по левому краю окна.	Lorem ipsum dolor sit amet, consectetuer adipiscing elit...
<img src="smile.jpg" align=**right**>	Выравнивает изображение по правому краю окна.	Lorem ipsum dolor sit amet, consectetuer adipiscing elit...

Наиболее популярные атрибуты **– left** и **right**, создающие обтекание текста вокруг изображения.

Пример 3. Вставка изображения с обтеканием

```
<html>
<body>
<p><img src="image/irgau.jpg" align="left">Текст, в который просто вставлена картинка. Задано расположение изображения у левого края страницы, текст обтекает справа.</p>
</body>
</html>
```

Браузер отобразит код так:

Текст, в который просто вставлена картинка. Задано расположение изображения у левого края страницы, текст обтекает справа.

Аналогичным образом можно задать обтекание рисунка другими способами, представленными в таблице.

Создание отступов при обтекании текста

Кроме способа выравнивания и обтекания текстом, для изображения можно задать поля отступов, которые не будут заняты текстом при обтекании. Задаются отступы двумя атрибутами: **vspace** - верхний и нижний, **hspace** - левый и правый. Значения задаются в пикселях (px). Рассмотрим пример с выравниванием изображения по левому краю и отступами слева и справа - 35px, сверху и снизу - 25px.

Пример 4. Задание отступов

```
<html>
<body>
<p>Текст в который просто вставлена
картинка. <img src="image/irgau.jpg" align="left"hspace="35" vspace="25">
Задано расположение изображения у левого края страницы, текст
обтекает справа.</p>
</body>
</html>
```

Результат выглядит следующим образом:

Создание рамки вокруг текста

В HTML можно задать рамку, определенной толщины, вокруг изображения. Делается это с помощью атрибута **border**, значение которого задается в пикселях. Единственный минус в том, что цвет рамки будет черным, а изменить его средствами HTML никак нельзя. Но можно заранее на изображении, в графическом редакторе, выполнить рамку произвольного цвета. Рассмотрим пример написания кода вставки изображения с рамкой.

Пример 5. Создание рамки вокруг текста

```
<html>
<body>
<p>Текст, в который вставлена картинка с
рамкой.<img src="image/irgau.jpg" align="middle" border="2">Задано
расположение по центру.</p>
</body>
</html>
```

В результате в браузере отобразится следующее:

Текст, в который вставлена картинка с рамкой.

Задано расположение по центру.

Изображения - ссылки

Изображения могут быть не только приемом web-дизайна, но и гиперссылками на другие web-страницы. Делается это точно так же, как с текстом, при помощи того самого тега **<A>**. В тег **<A>** просто включается тег нужного изображения. Лучше всего к такой ссылке (как и к любой гиперссылке) добавить тег **title**, так как это дает дополнительную информацию.

Пример 6. Вставка изображения-ссылки

```
<html>
<body>
<a href="http://www.igsha.ru/" title="Логотип сайта igsha.ru"><img
src="image/irgau.jpg" border="0"></a >
</body>
</html>
```

В браузере отобразится логотип ИрГАУ, по нажатию которого открывается сайт вуза:

При наведении курсора к изображению появляется всплывающая подсказка «Логотип сайта igsha.ru». По умолчанию изображения, содержащие гиперссылку, отображаются в рамке фиолетового цвета. Для того чтобы она не отображалась значение атрибута border необходимо установить равным нулю (**border="0"**).

Альтернативный текст

В каждом браузере есть функция отключения изображений. Пользователь, использующий такую функцию, может видеть описание того что

представляет собой изображение изображения. Это описание (или альтернативный текст) указывается в теге **alt**.

Пример 7. Создание альтернативного текста

```
<html>
<body>
<img src="image/irgau.jpg" border="0" title="Логотип сайта igsha.ru"
alt="Изображение логотипа сайта igsha.ru">
</body>
</html>
```

Увидеть альтернативный текст можно выключив отображение изображений в браузере.

Изменение размера изображения

Размеры изображения задаются атрибутами **width** - ширина и **height** - высота, значения задаются как в пикселях, так и в процентах от ширины экрана.

Пример 8. Изменение размера изображения

```
<html>
<body>
<img src="image/irgau.jpg" border="0" title="Логотип сайта igsha.ru"
alt="Изображение логотипа сайта igsha.ru" width="100" height="100">
</body>
</html>
```

Некоторые рекомендации по вставке изображений на web-страницу.

1. Старайтесь не использовать слишком большие размеры файла изображения, так как это влияет на скорость загрузки страницы.
2. Атрибут **ALT** имеет очень важное значение, рекомендуется добавлять его для каждого тега **IMG**. Содержание текстового сообщения должно очень точно описывать изображение, причем кратко.
3. Изображения на web-странице должны соответствовать текстовому содержанию.

Задание

1. Предположим, у Вас есть файл изображения mypicture.gif. Включите в HTML-страницу это изображение:

```
<img src="mypicture.gif">
```

Заметим, что в этом случае файл mypicture.gif должен находиться в том же самом каталоге (папке), где находится ваша страница.

2. Загрузите страницу в браузер и посмотрите, что получилось.

3. Включите в HTML-страницу изображение, которая находится в специальном каталоге (папке) для файлов изображений. Обычно, такой каталог называется images и он является подкаталогом основного каталога, где находятся файлы страниц.

Создайте такой каталог и включите в Вашу страницу следующий тэг:

```
<img src="images/mypicture.gif">
```

4. Загрузите страницу в браузер и посмотрите, что получилось.

5. Создайте каталог images2, который находится на том же уровне, что и каталог с Вашей страницей. Включите в Вашу страницу следующий тэг:

```
<img src="../images2/mypicture.gif">
```

6. Загрузите страницу в браузер и посмотрите, что получилось.

7. Измените размеры изображения, включенного в Вашу HTML-страницу:

```
<img src="mypicture.gif" width="x" height="y">,
```

где x – ширина изображения в пикселях, а y – высота изображения (также в пикселях).

8. Сделайте рамку вокруг изображения:

```
<img src="mypicture.gif" width="x" height="y" border="2">
```

Заметьте, что значение атрибута **border** есть толщина рамки.

9. Загрузите страницу в браузер и посмотрите, что получилось.

Задания для самостоятельной работы

1. Скопируйте любое изображение, расположенное в папке images в свою папку, внедрите его в текст из лабораторной работы №1 используя относительные ссылки и выровняйте его относительно текста согласно варианту.

№ варианта	Способ выравнивания
1	верхняя граница изображения выравнивается по самому высокому элементу текущей строки

2	выравнивание середины изображения по базовой линии текущей строки
3	выравнивание середины изображения посередине текущей строки
4	выравнивание изображения по базовой линии текущей строки
5	выравнивание нижней границы изображения по окружающему тексту
6	выравнивание изображения по левому краю окна
7	выравнивание изображения по правому краю окна
8	выравнивание верхней границы изображения по самому высокому текстовому элементу текущей строки
9	выравнивание нижней границы изображения по окружающему тексту
10	выравнивание середины изображения посередине текущей строки

2. Добавьте к рисунку альтернативный текст, с произвольным содержанием.

3. Добавьте в изображение ссылку на страницу с информацией о вашем факультете (используйте официальный сайт вуза).

4. Измените цвет и толщину рамки вокруг изображения.

Задание №5. Создание таблиц

Таблица состоит из строк и столбцов ячеек, которые могут содержать текст и рисунки. Обычно таблицы используются для упорядочения и представления данных, однако возможности таблиц этим не ограничиваются. Спомощью таблиц удобно верстать макеты страниц, расположив нужным образом фрагменты текста и изображений.

Для добавления таблицы на веб-страницу используется тег-контейнер **TABLE**. Таблица должна содержать хотя бы одну строку и колонку.

Для добавления строк используются теги **<tr>** и **</tr>**. Чтобы разделить строки на колонки применяются теги **<td>** и **</td>**. Тэги **<th>** и **</th>** иногда заменяют **<td>** и **</td>** когда необходимо выровнять текст по центру и сделать его полужирным.

Параметры таблицы

Для изменения вида и свойств таблицы используется множество атрибутов, которые добавляются в теге **TABLE**.

<table атрибут1=... атрибут2=...>

Описание атрибутов таблицы и их свойств описано ниже.

Свойство	Значение	Описание	Пример
align=	Left Right Center	Выравнивание таблицы	align=center
background=	URL	Фоновый рисунок	background=pic.gif
bgcolor=	#rrggbb	Цвет фона таблицы	bgcolor=#FF9900
border=	n	Толщина рамки в пикселах	border=2
bordercolor=	#rrggbb	Цвет рамки	bordercolor=#333333
bordercolordark=	#rrggbb	Тень рамки	border-color dark=#f0f0f0
cellpadding=	n	Расстояние между ячейкой и ее содержимым	cellpadding=7
cellspacing=	n	Дистанция между ячейками	cellspacing=3
nowrap		Запрещает переносы строк в тексте	<table nowrap>
frame=	Void Above Below Lhs Rhs Hsides Vsides Box	Задание типа рамки таблицы	frame=hsides
valign=	Top Bottom	Выравнивание по высоте	valign=top
width=	n n%	Минимальная ширина таблицы, можно задавать в пикселах или процентах	width=90%
height	n n%	Минимальная высота таблицы, можно задавать в пикселах или процентах	height=18

Примечание

1. Таблица, если не указано особо, всегда выравнивается по левому краю.

2. Если ширина таблицы не указана, она подгоняется под содержание ячеек.

Задание

Создать 3 таблицы, используя приведенный ниже листинг

1.1 Создание таблицы

```
<html>
<head>
<title>таблица</title>
```

```
</head>
<body>
<table border="2" align=center>
<tr>
<td colspan=2 align=center>
<b>Заголовок таблицы</b>
</td>
</tr>
<tr>
<td align="center">
Первая ячейка первой строки
</td>
<td align="center">
Вторая ячейка первой строки
</td>
</tr>
<tr>
<td align="center">
Первая ячейка второй строки
</td>
<td align="center">
Вторая ячейка второй строки
</td>
</tr>
</table>
</body>
</html>
```

Тег **<TABLE>** задает таблицу. Атрибуты **border="2"** и **align=center** задают, соответственно, размер границ таблицы и выравнивание ее по центру страницы Тег **<TR>** задает строку таблицы Тег **<TD>** задает ячейку таблицы

Атрибут **colspan=n** объединяет n ячеек по горизонтали (по столбцам) В результате получится следующая таблица, состоящая из двух столбцов и двух строк:

Заголовок Таблицы.	
Первая ячейка первой строки	Вторая ячейка первой строки
Первая ячейка второй строки	Вторая ячейка второй строки

1.2. Сложная таблица

```
<html>
<head>
<title>сложная таблица</title>
</head>
<body>
<table border="1" width="75%" align=center>
<tr>
<td width="66%" colspan="2">
<p align="center">Две ячейки, объединенные по горизонтали
</td>
</tr>
<tr>
<td width="33%" rowspan="2" valign="middle">Две ячейки, объединенные
по вертикали
</td>
<td width="33%">
по левому краю
</td>
</tr>
<tr>
<td width="33%">
<p align="right">по правому краю
</td>
</tr>
</table>
</body>
</html>
```

Атрибут **colspan=n** объединяетn ячеек по горизонтали (по столбцам)
Атрибут **rowspan=n** объединяетn ячеек по вертикали (по строкам)
Атрибут **valign="middle"** выравнивает текст в ячейке по центру ячейки по вертикали

В результате получается следующая таблица:

Две ячейки, объединенные по горизонтали	
Две ячейки, объединенные по вертикали	по левому краю
	по правому краю

1.3. Более сложная таблица

```
<html>
<head>
<title>УЧЕБНАЯ ТАБЛИЦА</title>
</head>
<body>
<table width="50%" border="6" cellspacing="6" cellpadding="20"
align="center" bordercolorlight="lime" bordercolordark="green"
bgcolor="#dfffdf"><thead bgcolor="aqua">
<tr><th colspan="3">УЧЕБНАЯ ТАБЛИЦА</th></tr>
</thead>
<tbody>
<tr>
<td width="33%">Это первая ячейка</td>
<td width="33%">Это вторая ячейка</td>
<td rowspan="3">А это три ячейки третьего столбца объединились в одну
большую</td>
</tr>
<tr>
<td colspan="2">Это единственная ячейка второй строки, объединяющая
оба столбца</td>
</tr>
<tr>
<TD> Это первая ячейка третьей строки</td>
<td width="33%">А это вторая ячейка третьей строки</td>
</tr>
</tbody>
<tfoot bgcolor="yellow">
<tr>
<td colspan="3" align="center">
<small>конец</small></td></tr>
</tfoot>
</table>
</body>
</html>
```

В результате получается следующая таблица:

Атрибут **CELLSPACING="6"** задает свободное пространство между ячейками таблицы

Атрибут **CELLPADDING="20"** задает свободное пространство между данными в ячейке и ее границами

Атрибут **BORDER COLORLIGHT="Lime"** задает цвет левого и верхнего углов таблицы

Атрибут **BORDER COLORDARK="Green"** задает цвет правого и нижнего углов таблицы

Атрибут **BGCOLOR="#DFFFDF"** задает цвет фона таблицы В

Задания для самостоятельной работы

Используя атрибуты объединения ячеек, изменения цвета фона и выравнивания текста создайте таблицу в HTML согласно образцу по варианту.

Вариант 1

Таблица. Вариант 1		
Это первая ячейка	Это вторая ячейка	Это две объеденные ячейки
Это первая ячейка второй строки	Это две объеденные ячеки второго столбца	
Это первая ячейка третьей строки		Это последняя ячейка

Вариант 2

Таблица. Вариант 2		
Это первая ячейка	Это вторая ячейка	Это третья ячейка
Это три объеденные ячейки		
Это первая ячейка третьей строки	Это две объеденные ячейки	

Вариант 3

Таблица. Вариант 3		
Это первая ячейка	Это две объеденные ячейки	Это третья ячейка
Это две объеденные ячейки первого столбца	Это вторая ячейка третьей строки	Это две объеденные ячейки

Вариант 4

Таблица. Вариант 4		
Это три объеденные ячейки первого столбца	Это вторая ячейка	Это третья ячейка
	Это две объеденные ячейки	
	Это вторая ячейка третьей строки	Это последняя ячейка

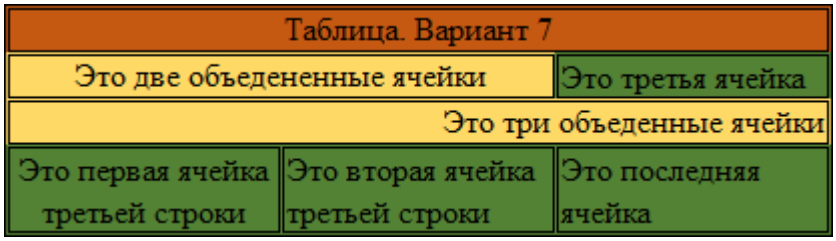
Вариант 5

Таблица. Вариант 5		
Это две объеденные ячейки первого столбца	Это вторая ячейка	Это третья ячейка
Это первая ячейка третьей строки	Это две объеденные ячейки второго столбца	Это две объеденные ячейки третьего столбца

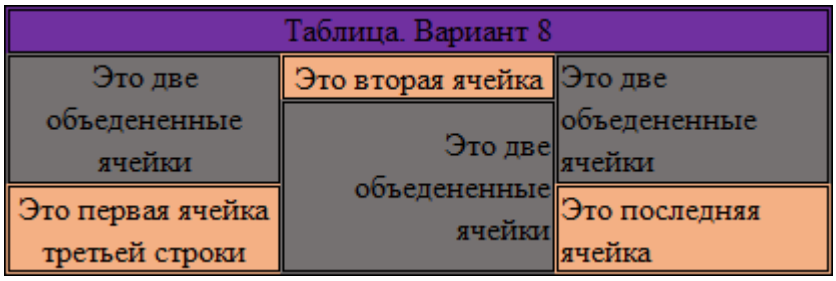
Вариант 6

Таблица. Вариант 6		
Это две объеденные ячейки	Это вторая ячейка	Это третья ячейка
	Это две объеденные ячеки второго столбца	Это третья ячейка второй строки
Это первая ячейка третьей строки		Это последняя ячейка

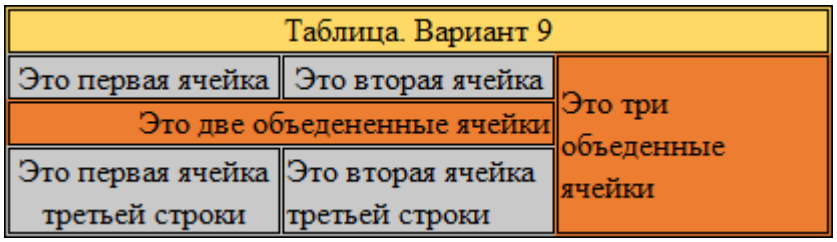
Вариант 7



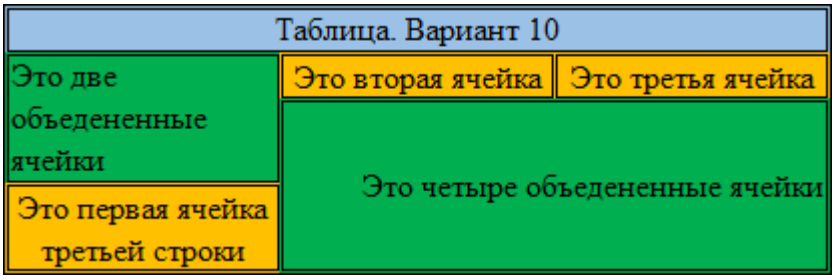
Вариант 8



Вариант 9



Вариант 10



Задание №6. Создание фреймов

Фреймы – это области окна браузера, в которые одновременно могут быть загружены разные страницы. Например, можно создать один фрейм для меню сайта, а другой фрейм будет использоваться для отображения информации, соответствующей каждому из пунктов меню.

Два вида тэгов используются для создания фреймов: **<frameset>** и <**frame**>. Первый из них определяет характеристики набора используемых фреймов, а второй – характеристики каждого фрейма.

Синтаксис тэга <**frameset**> следующий:

```
<frameset rows|cols="a,b,c,..." border=n frameborder="1|0" framespacing=n>
...
</frameset>
```

Атрибуты **rows** и **cols** определяют, как должны располагаться фреймы в этом наборе фреймов: строками или столбцами. Если необходимо и то и другое, следует использовать вложенные наборы фреймов. Каждое из значений "a, b, c, …" может быть следующим:

- **n** - высота/ширина строки/столбца, заданная в пикселях.
- **n%** - высота/ширина строки/столбца, заданная в процентах от высоты/ширины родительского окна или фрейма.
- ***** - высота/ширина строки/столбца определяется доступным пространством.
- **n*** - строка/столбец займет в n раз больше пространства, чем строка/столбец, определенные с *.

Атрибут **border** определяет ширину рамки фреймов.

Атрибут **frameborder** определяет, должны ли фреймы иметь рамку. Если указано

frameborder="0", то рамки не будет.

Атрибуты **framespacing** и **border** (используются для разных браузеров) определяет ширину рамки фреймов. Полезно их использовать одновременно.

Синтаксис тэга <**frame**>следующий:

```
<frame src="framesource" name="framename"
scrolling="yes|no|auto" frameborder="1|0"
noresize marginwidth="n" marginheight="n">
...
</frame>
```

Атрибут **src** определяет HTML-страницу, загружаемую во фрейм. Например, <**frame src="main.html"**>.

Атрибут **name** задает имя фрейма.

Атрибут **scrolling** определяет, должен ли фрейм иметь полосы прокрутки содержания.

Атрибут **frameborder** определяет, должен ли фрейм иметь рамку. Если указано

frameborder="0", то рамки не будет.

Атрибут **noresize** определяет, что пользователь не сможет динамически менять размер фрейма.

Атрибуты **marginwidth** и **marginheight** определяют расстояние (в пикселях) между содержимым фрейма и рамкой.

Часто необходимо загрузить во фрейм страницу, а гиперссылка для выполнения этой операции расположена в другом фрейме. В этом случае Вы должны использовать в тэге <**a**> атрибут **target** и указать в качестве его значения имя фрейма. Например, для того, чтобы загрузить во фрейм **main** из фрейма **menu** страницу **services.htm**, разместите во фрейме **menu** следующую ссылку:

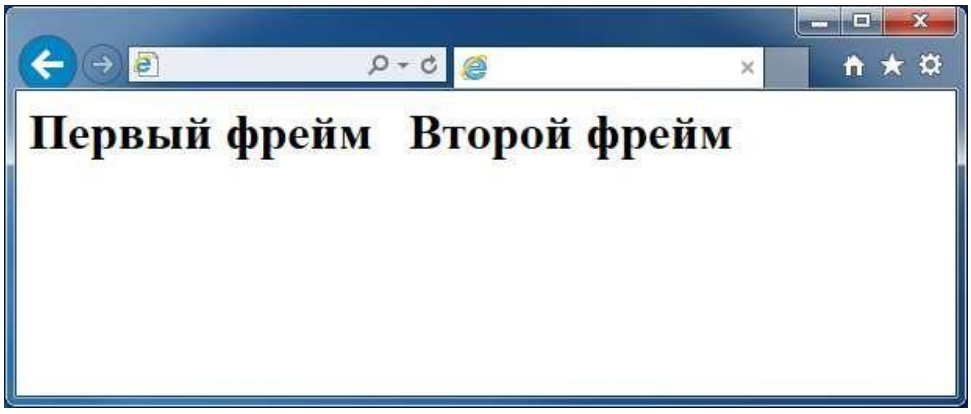
```
<a href="services.html" target="main">Наши услуги</a>
```

Ниже перечислены специальные значения, которые можно использовать для атрибута **target**:

- **_parent** - страница будет загружена в текущий фрейм набора фреймов предыдущего уровня вложенности;
- **_top** - страница будет загружена непосредственно в окно браузера;
- **_blank** - страница будет загружена в новое окно браузера;
- **_self** - страница будет загружена в тот же самый фрейм, где расположена ссылка на нее.

Пример 1. Создание простого фрейма

```
<html>
<frameset cols="40%, 60%" border=0>
<frame src="menu.html" name="menu">
<frame src="main.html" name="main">
</frameset>
</html>
```



В приведенном примере присутствует объявление двух фреймов, которые будут располагаться вертикальными полосами и занимать соответственно 30 и 70 процентов рабочей области. Вертикальное расположение устанавливается атрибутом **rows="..."**, а для горизонтальных полос используют атрибут **cols="..."**. Параметр **border="..."** определяет границу между фреймами. Определение документа, изначально загружаемого

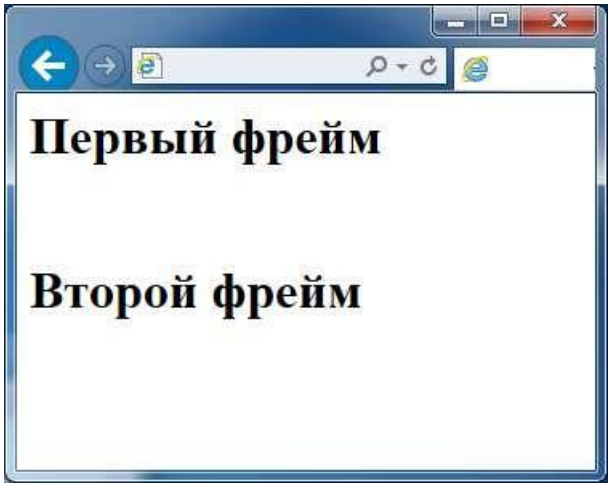
при открытии этого файла (этот документ является HTML-страницей.), задается атрибутом тега **<FRAME> src="...",** при этом необходимо указать атрибут **name="...",** позволяющий задать "имя" созданной области в виде последовательности латинских букв и цифр, использованной как значение этого атрибута. Это имя можно использовать, чтобы загружать новые документы в ранее созданную область. Для этого в тег **<A>,** определяющий гиперссылку, необходимо добавить атрибут **target="...",** значение которого совпадает с ранее определенным именем области. При переходе по данной гиперссылке новый документ загрузится в указанный фрейм. Например, предположим, что начальная страница Web-узла состоит из двух фреймов: слева располагается навигационная панель, а справа - текущая страница. Если правой области присвоено имя, используемое во всех ссылках, имеющихся в левой области, то щелчок на любой ссылке навигационной панели приведет к обновлению информации в соседней области, оставляя навигационную панель без изменений. В данном документе находятся только ссылки на другие (существующие) HTML-документы, которые будут загружены браузером при открытии страницы с фреймами. Файл **menu.html** имеет следующую структуру:

```
<html>
<head>
<title></title>
</head>
<body>
<h1>Первый фрейм</h1>
</body>
</html>
```

Файл main.html также имеет указанную выше структуру. Следует сказать, что оба этих файла должны располагаться в том же каталоге, где расположен файл с фреймами.

Задание 1.

Расположить эти фреймы по вертикали, используя атрибут **rows="..."**. В результате у вас должно получиться следующее:



Не все браузеры могут поддерживать фреймы. Для таких браузеров предусмотрено использование тэгов <**noframes**> и </**noframes**>, которые должны быть размещены перед тэгом </**frameset**>. Например, так:

```
...
<noframes>
<center><b>spoil yourself -get a new browser!</b></center>
</noframes>
</frameset>
```

Текст, размещенный внутри указанных тэгов, будет игнорироваться браузерами, поддерживающими фреймы.

Задание 2.

1. Создайте набор фреймов, состоящий из меню в левой части экрана и фрейма для показа содержательных страниц – в правой. Например, так:

```
<html>
<head>
<title>Пример набора из двух фреймов</title>
</head>
<frameset cols="140,*">
<frame name="menu" src="menu.htm">
<frame name="main" src="welcome.htm">
</frameset>
</html>
```

2. Загрузите страницу в браузер и посмотрите, что получилось.

3. Измените значения атрибутов тэгов <frameset> и <frame> и посмотрите, как изменяется внешний вид фреймов, загружая страницу в браузер.

4. Создайте страницы menu.htm и welcome.htm для более наглядной работы этого примера.

5. Создайте набор фреймов, состоящий из двух меню: в верхней и правой части экрана, а также фрейма для показа содержательных страниц в левой нижней части экрана. Например, так:

```
<html>
<head>
<title>Пример набора из двух фреймов</title>
</head>
<frameset cols="*,150">
<frameset rows="104,*">
<frame name="top" src="top.htm" noresize scrolling=no
marginheight=5 marginwidth=5>
<frame name="main" src="welcome.htm" noresize scrolling=auto
marginheight=5 marginwidth=5>
</frameset>
<frame name="right" src="right.htm">
</frameset>
</html>
```

6. Создайте страницы top.htm и right.htm для более наглядной работы этого примера.

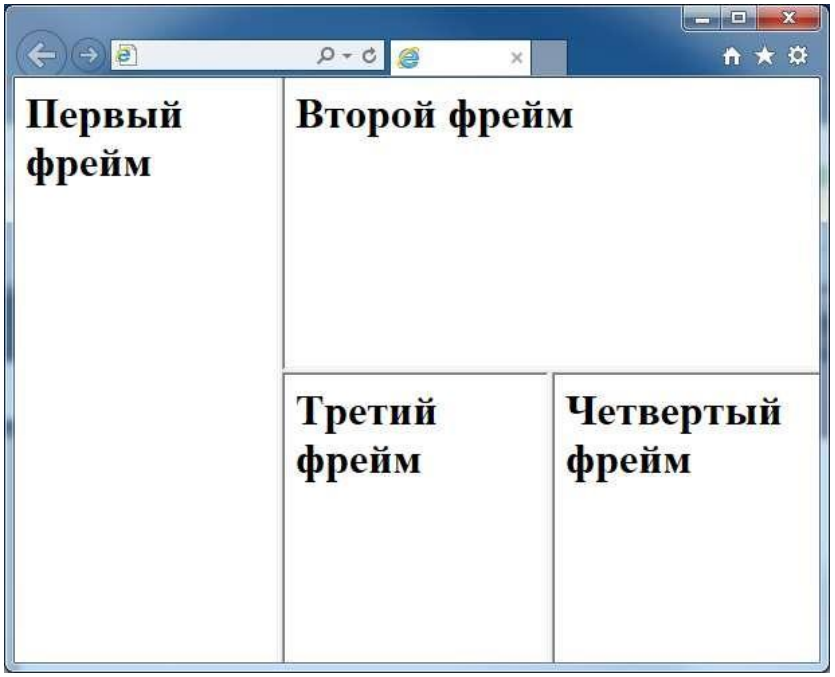
Задания для самостоятельной работы

Задание 1.

Распространенное явление - комбинация вертикальных и горизонтальных фреймов.

```
<frameset cols="*, 55%"> символ * означает все оставшееся место
<frame src="homepage.html" name="frame1">
<frameset rows="15%, 15%, 70%">
<frame src="menu.html" name="frame2">
<frame src="menu2.html" name="frame3">
<frame src="menu3.html" name="frame4">
</frameset>
</frameset>
```

Используя вложенные теги **FRAMESET**, разбейте страницу следующим образом:



Задание 2.

Используя атрибут **target="...",** сделать в фреймах гиперссылки, по нажатию на которые в эти же или в другие фреймы произойдет загрузка других страниц, полученных в предыдущих лабораторных работах.

Задание №7. Каскадные таблицы стилей CSS

Каскадные таблицы стилей или CSS (от английского Cascading Style Sheets) являются следствием дальнейшего развития HTML и дают нам возможность перейти на следующий уровень представления информации. Таблицы стилей позволяют разделить смысловое содержимое странички и его оформление.

Для подключения таблиц стилей CSS мы можем воспользоваться одним из 3-х предлагаемых методов:

- внешний файл;
- описание, встроенное в тег;
- описание в секции заголовка.

Описание, встроенное в тег (Inline-описание)

При помощи дополнительного атрибута **style** мы можем определить нужные нам стилевые атрибуты в любом теге.

```
<p style="color:red; text-align:center;">
```

```
Этот текст переопределен стилем
</p>
```

Это самый легкий способ, и действует он в пределах лишь одного тега. Этот способ схож с прямым описанием внешнего вида при помощи тега **<font>**.

Описание в секции заголовка

В отличие от описания, встроенное в тег действие этого метода распространяется на всю страничку. Определение стилей происходит при помощи классов, которые представляют собой списки с определением всех необходимых параметров оформления.

При использовании этого метода описание стилей необходимо разместить в секции заголовка:

```
<head> ....
<style type="text/css">
<!--
.header {
text-align :center;
font-size : 27pt;}
.red {color : red; }
-->
</style>
</head>
```

Теперь эти стили можно применять в любом месте html-кода. Для этого используется следующая конструкция:

```
<p class=header>Этот текст написан стилем header<p>
<p class=red>Этот текст написан красным цветом<p>
```

Кроме определения новых классов мы также имеем возможность переопределять стандартные теги. Например, тег :

```
<style type="text/css">
<!-- p { text-align : center; font-size :12pt;}
--> </style>
```

Теперь весь текст, заключенный в теги ** ** и ** **, будет выглядеть так, как определено данным стилем. Это очень удобно и позволяет легко адаптировать уже существующие странички к использованию стилей. Кроме

того, это несколько уменьшает объем файла за счет отсутствия лишних атрибутов **class**.

Вынесение описания стилей во внешний файл

Диапазон его воздействия простирается на все файлы, в которые включено описание. В случае, если нам потребуется изменить внешний вид сайта, то будет достаточно скорректировать лишь один этот файл.

Для начала создается стилевой файл с описанием всех нужных нам классов (mystyle.css):

```
.header { text-align : center; font-size : 27pt;}
.red { color :red; }
p { text-align : center; font-size : 12pt;}
```

А потом ссылка на него внедряется в документ при помощи тега <link>:

```
<head>..... <link rel="stylesheet" type="text/css"
href="css/mystyle.css" title="MyStyleSheet"> .....</head>
```

Это самый удобный способ, и для основной таблицы стилей рекомендуется пользоваться именно им.

Каскадность стилей

Каскадность заключается в том, что стили могут переопределяться. Приведенный выше список способов внедрения стилей соответствует порядку переопределения. Нижерасположенный способ может переопределять вышерасположенный.

Например, мы определили во внешнем стилевом файле, что текст в теге **<p>** должен быть написан при помощи шрифта высотой 10 пунктов. Но если в заголовке странички мы дополнительно укажем, что тот же текст в теге **<p>** должен быть написан шрифтом в 12 пунктов, то текст будет выведен именно таким кеглем - т.е. стиль в заголовке странички переопределил стиль во внешнем файле.

Синтаксис CSS

Описание каждого класса делается при помощи конструкции, подобной этой:

```
.small { font-size: 9pt; }
```

Сначала указывается имя класса - оно может быть произвольным, но желательно все-таки давать осмысленное название. Далее, в фигурных скобках

перечисляются все необходимые параметры для данного класса. Параметры отделяются друг от друга точкой с запятой. Вот еще один пример, в котором используется более длинное описание:

Заметьте, что имя класса начинается с точки и таким образом определяет универсальный класс, т.е. такой, который может быть применен к любому тегу. И делается это при помощи следующей конструкции:

```
<p class=small>Накладываем стиль на этот текст</p>
```

Существуют универсальные классы и, так называемые, теговые классы:

```
p.small { font-size: 9pt; }
```

Класс, определенный таким образом, сработает только в том теге, для которого он предназначен, а для всех остальных будет проигнорирован.

Мы можем определять параметры не только для одного тега, но и сразу для нескольких. Для этого в определении стиля достаточно перечислить их через запятую:

```
p, td { font-size: 9pt; color:green;}
```

Такой прием называется группировкой, и в данном случае мы определили и для **<p>**, и для **<td>** одинаковый размер и цвет текста.

В случае переопределения существующих тегов, в описании стиля можно указывать не все параметры, а лишь те из них, которые мы хотим изменить. Все остальные параметры примут значения по умолчанию, которые для разных тегов различны.

Псевдоклассы

В CSS есть такое понятие как псевдокласс. В отличие от обычного класса, действие псевдокласса распространяется не на весь текст, к которому применен данный стиль, а лишь на его часть и возможно лишь в определенном состоянии. Чтобы было понятнее, давайте разберем эффект, при котором ссылки подчеркиваются лишь при наведении на них курсора. Эффект достаточно распространенный, и его можно наблюдать в том числе и на этом сайте. Вот фрагмент таблицы стилей, который позволяет достигать вышеописанного эффекта:

```
a { text-decoration: none; }
a:hover { text-decoration: underline;}
```

Верхняя строчка - это переопределение стандартного тега **<a>**, которое запрещает подчеркивать ссылки, а вот нижняя - это определение стиля для псевдокласса **hover**, который описывает стиль ссылки в момент, когда курсор находится над ней.

А вот и другой пример псевдокласса - определение буквицы вначале абзаца:

```
p:first-letter { font-size: 200%; font-weight: bold; }
```

Заметьте, что и в том, и в другом случае действие стиля распространяется либо на определенное состояние (пользователь собирается щелкнуть по ссылке), либо на фрагмент текста (изменяется только первая буква абзаца). В этом и заключается смысл псевдоклассов.

Основные атрибуты CSS

Все атрибуты, используемые для определении стиля, условно можно разделить на несколько больших групп: управляющие видом шрифта (гарнитура, кегль, цвет, наклон, жирность...) управляющие форматированием абзаца (выравнивание, интерлиньяж, расстояние между словами, отступ красной строки...) управляющие свойствами блока (отступы слева-сверху-справа-снизу, местоположение блока на страничке, видимость блока...) другие, которые не вписываются ни в одну из перечисленных выше групп (цвет ссылок странички, изменение внешнего вида курсора...) Рассмотрим подробнее атрибуты, используемые для управления внешним видом текста и форматирования абзацев - как наиболее часто употребляемые.

Свойства шрифта

Свойство	**Значение**	**Описание**	**Пример**
font-family	имя шрифта	Задает список шрифтов	p {font-family: Arial, serif}
font-style	Normal Italic Oblique	Нормальный шрифт Курсив Наклонный шрифт	p {font-style: italic}
font-variant	Normal Small-caps	Нормальный шрифт Капитель (особые прописные буквы)	p {font-variant: small-caps}
font- weight	Normal Lighter Bold Bolder 100-900	Нормальная жирность Светлое начертание Полужирный Жирный 100-светлый	p {font-weight: bold}

		шрифт, 900-самый жирный	
font-size	Размер шрифта Normal Pt Px %	 нормальный размер пункты пикселы проценты	 font-size: normal font-size: 12pt font-size: 12px font-size: 120%

Замечание:

Когда размер шрифта задается абсолютными значениями, т.е. указывается конкретное значение шрифта в пунктах или пикселах, то изменить эту величину с помощью опции браузера *Вид* | *Размер шрифта* невозможно. Если шрифт установлен слишком мелким, то исправить этот недостаток читателю простыми средствами не представляется возможным. Поэтому лучше задавать размер шрифта в процентах (Пример 1).

Пример 1. Задание свойств шрифта с помощью CSS

```
<html>
<style>
h1 {font-family: Arial, Helvetica, Verdana, sans-serif; font-size: 150%; font-weight: light }
</style>
<body>
<h1>Заголовок</h1>
Обычный текст
</body>
</html>
```

Ниже приведен результат примера 1.

Заголовок

Обычный текст

В таблице Примера 2 даны некоторые атрибуты и результат их применения.

Пример 2. Результат использования различных атрибутов шрифтов

Пример	**Пример**	Пример	ПРИМЕР	*Пример*
font-family: Verdana, sans-serif; font-size: 120%; font-weight: light	font-size: large; font-weight: bold	font-family: Arial, sans-serif; font-size: x-small; font-weight: bold	font-variant: small-caps	font-style: italic; font- weight: bold

Свойства текста

Кроме изменения атрибутов шрифтов, можно управлять и свойствами всего текста. Значения свойств приведены в таблице.

Свойство	Значение	Описание	Пример
line-height	Normal Множитель Точно %	Интерлиньяж (межстрочный интервал)	line-height normal line-height 1.5 line-height 12px line-height120%
text-decoration	None Underline Overline Line-through Blink	Убрать все оформление Подчеркивание Линия над текстом Перечеркивание Мигание текста	text-decoration: none
text-transform	None Capitalize Uppercase Lowercase	Убрать все эффекты Начинать С Прописных ВСЕ ПРОПИСНЫЕ все строчные	text-transform: capitalize
text-align	Left Right Center Justify	Выравнивание текста	text-align: justify выравнивание по ширине
text- indent	Точно %	Отступ первой строки	text-indent:15px; text-indent:10%

Ниже, в Примере 3 приведены некоторые атрибуты текста и результат их применения.

Пример 3. Результат использования различных атрибутов текста

Пример: И Это Все О Нем	Пример: текст по центру	Пример: Это не ссылка, а просто текст	Пример: отступ первой строки	Пример: полуторный межстрочный интервал
text-transform: capitalize	text-align: center	text-decoration: underline	text-indent: 20px	line-height: 1.5

Границы и рамки

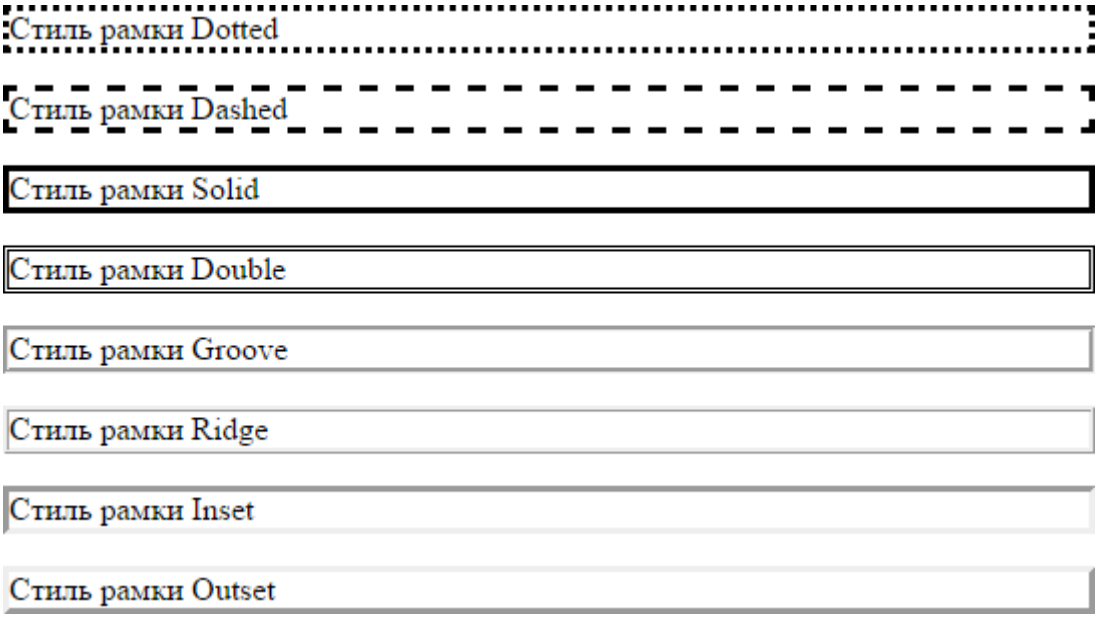
Спецификация CSS описывает несколько свойств, с помощью которых можно создавать границу вокруг различных элементов и управлять ее видом. Границы - одна из наиболее слабых сторон CSS, т.к. браузеры содержат большое количество ошибок и по-разному интерпретируют атрибуты. Старшие версии браузеров отображают рамки вокруг элементов более корректно.

Свойство	**Значение**	**Описание**	**Пример**
padding-top **padding-right** **padding-bottom** **padding-left** **padding**	Значение %	Отступ от границы элемента до его содержимого	table {padding: 15px 15px}
border-top-width **border-right-**	Thin Medium Thick	Ширина границы	P {border-top-width: 4px}
width **border-bottom-width** **border-left-width** **border-width**	Значение		
border-color	цвет	Цвет границы	P {border-color: red}
border-style	None Dotted Dashed Solid Double Groove Ridge Inset Outset	Стиль рамки	table {border-style: double}

border-top **border-right** **border-bottom** **border-left**	border-top-width border-style цвет	Определяет толщину, стиль и цвет каждой границы	table {border-top: solid 4px red; border-left: solid 4px blue}
border	см. выше	Задает толщину, стиль и цвет рамки	table {border: solid 4px red}

Типы рамок

Для управления видом рамки предоставляется восемь значений атрибута **border-style.** Результат их действия представлен на рисунке.



Пример 4. Оформление текста в рамке

```
<html>
<body>
<p style="color: yellow; background-color: deepskyblue; text-decoration: underline; text-transform: uppercase; border: pink inset 25; padding: 20; font-size: larger; line-height: 40px; text-align: center">Пример оформления текста в рамке с помощью CSS</p>
</body>
</html>
```

Атрибут **style="..."** задает стилевое оформление абзаца.

Атрибут **color: yellow** задает цвет текста.

Атрибут **background-color: deepskyblue** задает цвет фона для абзаца.

Атрибут **text-decoration: underline** задает подчеркивание для текста.

Атрибут **text-transform: uppercase** задает режим заглавных букв для текста.

Атрибут **border: pinkinset 25** задает рамку вокруг абзаца, соответственно, розовую выпуклую толщиной 25 пикселей.

Атрибут **padding: 20** задает отступ от границы элемента до его содержимого.

Атрибут **font-size: larger** задает размер шрифта

Атрибут **line-height: 40px** задает межстрочный интервал.

Атрибут **text-align: center** задает выравнивание текста внутри абзаца по центру.

Единицы измерения в CSS

В свойствах, которым требуется указание размеров, можно использовать несколько способов для их задания:

- относительный размер в процентах (%)
- относительный размер при помощи словесного описания (larger, smaller, xx-small, x-small, small, medium, large, x-large, xx-large)
- абсолютный размер в типографских единицах - размер может задаваться в пунктах (pt), пиках (pc), пикселях (px), средней шириной буквы "m" (em), средней шириной буквы "x" (ex)
- абсолютный размер в стандартных единицах длины - размер может задаваться в сантиметрах (cm), миллиметрах (mm), дюймах (in) абсолютный в пикселях (px)

Задание цвета в CSS

Цвет для тех свойств, где это нужно, может быть определен одним из трех способов:

- при помощи названия цвета: yellow, red, green, grey и т.д.;

- шестнадцатеричным заданием цвета в формате #RRGGBB: #ff0000, #883490, #ffffff,..;
- десятичным заданием составляющих цвета в формате rgb(red, green, blue): rgb(255,0,0), rgb(100,23,78),..;

Пример 5. Различные описания таблицы стилей

```
.epigraph {
   font-size: 12pt;
   font-style: italic;
   text-align: right;
   color: rgb(127,127,0);
}
```

```
p.big {
   font-size: 16px;
   font-weight: bold;
   color: #ff0000;
}
```

```
.menu {
   font-weight: bold;
   font-size: 9pt;
   font-family: arial, helvetica, sans-serif;
}
```

```
a:hover {
   color: #b63a3a;
   text-decoration: none;
}
```

Задание 1

Создать в графическом редакторе Paint небольшой рисунок, который будет использоваться в качестве маркера списка. Создать маркированный список со своим маркером, используя стилевое свойство **LIST-STYLE-IMAGE: url(имя_рисунка)**.

Например, <UL style="LIST-STYLE-IMAGE: url(star2.jpg)">

Задание 2

Скопируйте исходный текст страницы и измените стили элементов согласно требованиям.

Данный абзац написан шрифтом Arial Black и имеет размер 20 пикселей.

Данный абзац написан курсивным шрифтом Courier New и имеет размер 24 пикселя и выровнен по ширине. Установите красную строку. Это текстовое заполнение. Это текстовое заполнение. Это текстовое заполнение. Это текстовое заполнение. Это текстовое заполнение. Это текстовое заполнение. Это текстовое заполнение. Это текстовое заполнение. Это текстовое заполнение. Это текстовое заполнение.

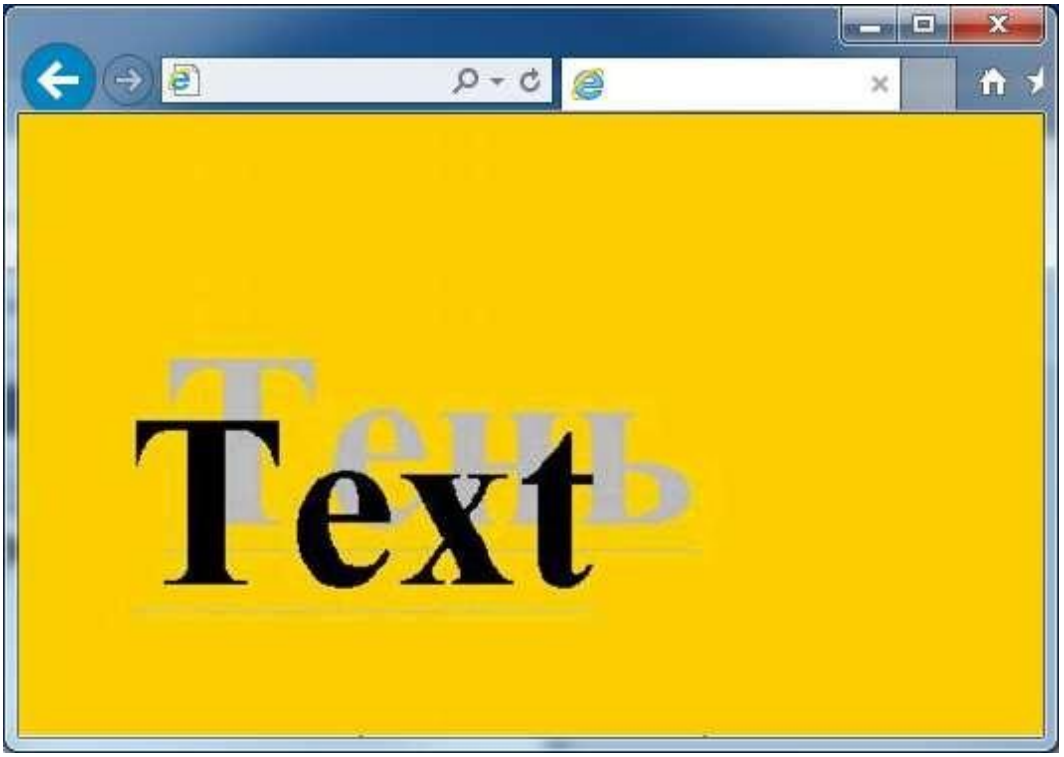
Данный абзац написан жирным шрифтом Verdana и имеет размер 10 пикселей.

Данный абзац написан шрифтом Georgia и имеет размер 2.5 em и отцентрирован

Данный абзац написан курсивным шрифтом Comic Sans MS и имеет размер 1.3 em.

Задание 3

Значения отступов вокруг объектов можно указывать как положительные, так и отрицательные. Таким образом, можно использовать данную возможность для наложения одного слоя текста на другой. Например, можно создать текст с тенью, без использования трехмерного изображения. Создайте два стиля, которые отличаются цветом и размером отступов вокруг них. Используя отрицательные значения отступов и подбирая нужное значение, можно добиться того, что верхний слой как бы наползает на предыдущий. В результате у вас должно получиться следующее:



используйте контейнер <DIV> и свойства margin-top и margin-left.

Задание для самостоятельной работы

Используя теги HTML и таблицу стилей CSS, создать тематический Web-сайт по вопросам дисциплины «Вычислительная техника и компьютерные сети». Тематику Web-сайта выбрать из таблицы по вариантам:

№ варианта	Тематика сайта
1	Принципы Фон-Неймана. Структура ЭВМ
2	История развития вычислительной техники. Поколения ЭВМ
3	Внутренние устройства компьютера
4	Периферийные устройства компьютера
5	Внешние устройства накопления информации
6	Память ЭВМ
7	Супер-ЭВМ
8	Технология Intranet. Протоколы и стандарты локальных сетей
9	Глобальные компьютерные сети. Сеть Internet
10	Уровни модели OSI

Информацию по выбранной теме найти в сети Интернет.

Требования к содержанию сайта:

1) на главной странице поместить название работы, информацию об авторе (ФИО, город, учебная группа, вуз), информацию о руководителе работы (ФИО, должность), название дисциплины, по которой выполнена работа;

2) информацию по теме поместить на разных страницах сайта в соответствии со структурой материала;

3) информационное наполнение сайта (контент) должно достаточно полно раскрывать тему, в то же время не следует перегружать сайт детальными подробностями – лучше сделайте соответствующую ссылку;

4) на отдельной странице привести гиперссылки на электронные ресурсы, которые были использованы при выполнении работы (сайт с указанием конкретного ресурса).

Требования к оформлению сайта:

1) организовать систему навигации по страницам сайта (гипертекстовое меню переходов на тематические разделы (страницы) сайта, гипертекстовые переходы со страницы на страницу («линейная навигация»), гипертекстовое оглавление отдельных страниц с большим по объему текстом, гипертекстовые переходы на начало страницы);

2) использовать фреймы для создания трехоконной структуры страницы, например: «шапка» страницы; меню; основная часть. Меню выполнить в виде гипертекстового перечня названий страниц сайта;

3) при оформлении «шапки» страницы использовать бегущую строку;

4) использовать однородную цветную заливку страницы, а также использовать картинку в качестве фона страницы;

5) при оформлении текста на страницах:

a) определить три своих собственных класса для оформления абзацев – назвать произвольно. В каждом классе определяются отступы спереди и сзади текста – в трех соответствующих классах определяются отступы разной величины. Использовать все данные классы для оформления разных абзацев текста.

b) использовать разный шрифт, цвет, размер, нижний и верхний индексы, выделение курсивом, полужирным, выравнивание слева, справа, по центру;

c) большой по объему текст разбить на пункты, подпункты, абзацы;

d) вставить списки (нумерованные и ненумерованные);

e) все ссылки сделать "мигающими" – то есть, чтобы при наведении на них курсором мышки, подчеркивание исчезало (использовать псевдокласс hover);

6) вставить графические объекты (рисунки, фотографии, графики) и оформить интерактивную подпись к ним. Использовать рисунок как гиперссылку. Использовать различные режимы обтекания картинки текстом. Все используемые рисунки поместить в отдельную папку.

Литература

1. Александровский А.Д. Создание Web-страниц с использованием Front Page 98 и JavaScript. - М.: ДМК, 1998 – 368 с.: ил.
2. Захаркина В. В. Каскадные таблицы стилей CSS: Учебное пособие. — СПб.: Ф-т филологии и искусств СПбГУ, 2007. — 44 с.
3. Кравченко С.В. Основы сайтостроения: Учеб. пособие / С. В. Кравченко. - Томск, 2012. — 300 с.
4. Мак-Дональд М. HTML5. Недостающее руководство: Пер. с англ. – СПб.: БХВ - Петербург, 2014. – 480 с.:ил.
5. Матросов А.В. HTML 4.0 / А.В. Матросов, А.О. Сергеев, Н.П. Чаунин. - СПб.: БХВ - Петербург, 2002. – 672 с.:ил.
6. Пауэлл Томас А. WEB - дизайн: Пер. с англ. СПб.: БХВ - Петербург, 2002. - 1024с.:ил.
7. Росс В.С. Создание сайтов: HTML, CSS, PHP, MySQL. Учебное пособие, ч. 1 — МГДД(Ю)Т, М.:2010 – 107 с.
8. Тубольцев М.Ф. Лабораторный практикум по Web-технологиям. Часть 1. Основы HTML технологий / М.Ф. Тубольцев, Н.П. Путивцева, И.В. Гурьянова, О.В. Немыкина. – Белгород: Изд-во БелГУ, 2003

Приложение

Краткий справочник по тегам HTML

Тег	Версия HTML	Описание
A	1.0	Ссылка
ADDRESS	2.0	Информация об авторе
APPLET	3.2	Javaапплет
AREA	3.2	Задание активной области для карты изображения
B	2.0	Жирный текст
BASE	2.0	URL, относительно которого ведется адресация ссылок
BASEFONT	3.2	Задание шрифта, размера и цвета текста по умолчанию
BGSOUND	-	Фоновая музыка
BIG	3.2	Увеличивает размер шрифта
BLINK	-	Мерцающий текст
BLOCKQUOTE	2.0	Создание цитаты
BODY	2.0	Тело документа
BR	2.0	Перевод строки
BUTTON	4.0	Кнопка
CAPTION	3.2	Заголовок таблицы
CENTER	3.2	Выравнивание элемента по центру окна браузера
CITE	2.0	Используется для выделения цитат
CODE	2.0	Обозначает фрагмент программного кода
COL	4.0	Колонка таблицы
COLGROUP	4.0	Служит для определения группой столбца таблицы
COMMENT	-	Комментарий игнорируемый браузером
DD	2.0	Описание списка определений
DEL	4.0	Отмечает текст как удаленный
DFN	3.2	Отмечает текстовый фрагмент как определение
DIR	2.0	Создание списка
DIV	3.2	Контейнер для задания стиля блока, используется также для создания слоев
DL	2.0	Создание списка определений
DT	2.0	Задание списка определений
EM	2.0	Выделение фрагмента текста
EMBED	-	Вставка внешнего объекта вHTML
FIELDSET	4.0	Группировка элементов форм
FONT	3.2	Изменение параметров шрифта
FORM	2.0	Добавление формы
FRAME	4.0	Определяет одиночный фрейм

FRAMESET	4.0	Разбивает окно браузера на фреймы
H1	2.0	Заголовок первого уровня
H2	2.0	Заголовок второго уровня
H3	2.0	Заголовок третьего уровня
H4	2.0	Заголовок четвертого уровня
H5	2.0	Заголовок пятого уровня
H6	2.0	Заголовок шестого уровня
HEAD	2.0	Блок для хранения служебной информации
HR	2.0	Горизонтальная линия
HTML	2.0	Указывает браузеру, что документ в формате HTML
I	2.0	Курсивное начертание текста
IFRAME	4.0	Плавающий фрейм
IMG	2.0	Вставка изображения
INPUT	2.0	Поле формы
INS	4.0	Отмечает текст как вставку
ISINDEX	2.0	Поиск по ключевым словам
KBD	2.0	Отмечает текст как вводимый пользователем с клавиатуры
LABEL	4.0	Название группы в форме
LAYER	-	Создание слоя
LEGEND	4.0	Создает рамку вокруг формы
LI	2.0	Элемент списка
LINK	2.0	Связь с другими документами
MAP	3.2	Карта-изображение
MARQUEE	-	Поле для перемещения текста
MENU	2.0	Создание списка
META	2.0	Метаинформация
MULTICOL	-	Многоколоночный текст
NOBR	-	Запрет перевода строки
NOEMBED	-	Альтернативный текст для встроенных объектов
NOFRAMES	4.0	Альтернативное содержание для браузеров не поддерживающих фреймы
NOLAYER	-	Альтернативное содержание для браузеров не поддерживающих слои
NOSCRIPT	4.0	Альтернативное содержание для браузеров не поддерживающих скрипты
OBJECT	4.0	Вставка внешнего объекта
OL	2.0	Нумерованный список
OPTION	2.0	Создание выпадающего списка
P	2.0	Параграф
PARAM	3.2	Задание значения параметру.

PLAINTEXT	-	Обычный текст
PRE	2.0	Текст пишется как есть, включая все пробелы
Q	4.0	Отмечает короткие цитаты в строке текста.
S	3.2	Зачеркнутый текст
SAMP	2.0	Отмечает текст как пример
SCRIPT	3.2	Область скрипта
SELECT	2.0	Элемент выпадающего списка
SMALL	3.2	Уменьшает размер шрифта
SPAN	4.0	Встроенный элемент для задания стиля фрагмента
STRIKE	3.2	Зачеркнутый текст
STRONG	2.0	Выделение фрагментов текста
STYLE	3.2	Указание стиля
SUB	3.2	Нижний индекс
SUP	3.2	Верхний индекс
TABLE	3.2	Таблица
TBODY	4.0	Тело таблицы
TD	3.2	Ячейка таблицы
TEXT AREA	2.0	Поле многострочного текста в форме
TFOOT	4.0	Нижний колонтитул таблицы
TH	3.2	Ячейка таблицы, которая является заголовком
THEAD	4.0	Верхний колонтитул таблицы
TITLE	2.0	Заголовок страницы
TR	3.2	Строка таблицы
TT	2.0	Моноширинный текст
U	3.2	Подчеркнутый текст
UL	2.0	Маркированный список
VAR	2.0	Отмечает имена переменных программ