САНКТ-ПЕТЕРБУРГКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Математико-механический факультет

О. Н. ИВАНЦОВА, И. Д. МИРОШНИЧЕНКО

ПРАКТИКУМ ПО УЧЕБНОЙ ДИСЦИПЛИНЕ «СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ» ЧАСТЬ ПЕРВАЯ

Учебно-методическое пособие

Санкт-Петербург
2020

Рецензенты:
доктор физ.-мат. наук, проф. О.А. Аксенова (Военно-морской политехнический институт)
доктор физ.-мат. наук, проф. Терехов А.Н. (Санкт-Петербургский гос. университет)

Рекомендовано к печати
Учебно-методической комиссией по УГСН 02.00.00 Компьютерные и информационные науки.
Протокол № 06/2-03-4 от 25.02.2020 г.

Иванцова О. Н., Мирошниченко И. Д.
Практикум по учебной дисциплине «Системное программирование» Часть первая. СПб, СПбГУ, 2020, – 31 с.

Методическое пособие «Практикум по учебной дисциплине «Системное программирование» Часть первая» (авторы: О.Н. Иванцова, И.Д. Мирошниченко), прежде всего, предназначено для обучающихся по учебному плану «Прикладная математика и информатика» в 4-ом семестре.

Данное пособие призвано помочь обучающимся в полном объеме выполнить все задания, рассматриваемые при обучении дисциплине «Системное программирование» на практических занятиях. Особенно полезно пособие окажется тем, кто не имеет значительного опыта создание web-страниц с использованием языка разметки текста HTML, каскадных таблиц стилей CSS, а также языка программирования JavaScript.

В пособии подробно рассмотрены задания, указаны способы их выполнения и приведены методические замечания. Пособие содержит большое количество наглядных примеров сценариев (в виде рисунков) и их кодов, которые призваны помочь обучающему успешно освоить дисциплину «Системное программирование».

Задания, посвященные языку HTML, включают в себя следующее: работа со шрифтовым и абзацным форматированием текста web-страницы; использование графики и мультимедиа; разнообразное использование гиперссылок и таблиц; описание создания фреймовых структур и навигационных карт.

Задания, связанные с изучением каскадных таблицы стилей CSS, позволяют обучающемуся использовать альтернативные методы для создания web-страниц, в отличие от методов, рассмотренных выше. В данном случае имеется возможность написания, а в дальнейшем и использования внешних файлов для формирования разнообразных web-страниц. Рассматривается более тонкая организация различных дополнительных элементов форматирования текста и дизайна web-страницы.

Тема, посвященная использованию языка программирования JavaScript для создания web-страниц, описывает простейшие сценарии, с помощью которых можно сделать web-страницу более интерактивной.

ПРЕДИСЛОВИЕ

Практикум по дисциплине «Системное программирование. Часть первая» предназначен для обучающихся второго курса направления «Прикладная математика и информатика» для овладения простыми средствами web-программирования, начиная с использования языка форматирования HTML и каскадной таблицы стилей CSS, а также языка программирования JavaScript.

Методические материалы первой части практикума будут полезны тем обучающимся, которые не имеют опыта web-проектирования. В практикуме предлагаются базовые методы и приёмы разработки статических и псевдодинамических проектов.

Практикум включает описание типовых практических заданий и их реализаций, а также заданий для самостоятельной работы.

Первая часть практикума сформирована таким образом, чтобы логически и содержательно-методически соответствовать рабочей учебной программе по дисциплине «Системное программирование», которая читается в 4 семестре периода обучения и рекомендуется к использованию в дальнейшем при изучении дисциплин, базирующихся на web-программировании.

Практикум включает описание практических заданий на языке HTML, практических заданий с использованием таблиц каскадных стилей CSS и практических заданий на языке JavaScript.

В соответствии с учебным планом практические занятия, соответствующие предлагаемому практикуму, приводят к получению и закреплению знаний, умений, владений обучающимися компетенцией ПКП-5, а именно, развитию способности проектировать и разрабатывать компоненты программного обеспечения на основе современных парадигм, технологий и языков.

Предлагаемое пособие представляет собой первую часть методических материалов для выполнения практических работ по дисциплине «Системное программирование» и предполагает разработку методических материалов второй части, которая будет содержать задания и рекомендации по разработке клиент-серверных проектов с использованием языка web-программирования PHP и языка разработки баз данных MySQL.

Предлагаемые методические материалы могут служить практическим материалом для преподавателей и обучающихся при подготовке планов практических занятий в вузах, а также в средних специальных инженерно-технических учебных организациях.

ВВЕДЕНИЕ

Основные сведения о работе со средствами создания web-страниц

Простейшими средствами создания web-страниц являются, прежде всего, язык форматирования HTML и каскадная таблица стилей CSS.

HTML[1] — стандартизированный язык разметки электронных документов в Интернет (или Интранет) сети. Язык HTML – декларативный (или описательный) язык, то есть средствами языка – тегами – пользователь определяет *где* и *что* должно быть размещено на web-странице, представленной выбранным браузером после обработки. Текст программы на HTML языке (код программы) обрабатывается транслятором, который представляет собой *интерпретатор* (то есть может фрагментарно анализироваться и сразу исполняться). Полученный в результате интерпретации форматированный текст отображается на экране стационарного компьютера, ноутбука или мобильного устройства.

Каждый браузер содержит свой интерпретатор кода, интерпретации тегов языка HTML в различных браузерах часто различаются, и даже какие-то из тегов могут не функционировать в тех или иных браузерах. HTML-страницы, как правило, передаются браузерам по протоколам HTTP или HTTPS в виде текста (с расширением *html* или *htm*) или с использованием шифрования.

Со времени появления первого упоминания об языке HTML[2] разработано несколько стандартов (и даже версий стандартов) этого языка. Долгое время наиболее употребительным был стандарт HTML, который не использовал конструкции (селекторы) каскадной таблицы стилей CSS[3].

Первые практические работы настоящего практикума будут использовать именно этот стандарт для упрощенного понимания структуры языка.

В стандарте HTML 4.0[4] (стандарт ISO 8879) были отмечены некоторые теги как устаревшие и не рекомендованные, но возможность их использования осталась (для совместимости версий). Это, например, относится к тегу **<*font*>**, определяющему свойства шрифта, который рекомендовано было не использовать, а вместо него применять ювелирные возможности селекторов языка разметки каскадной таблицы стилей CSS.

Язык HTML5[5] предназначен для лучшего представления семантики различных специальных страниц (для форумов, поисковых систем, онлайн-магазинов), является расширением HTML особыми тегами, которые не очень удачно вписываются в стандарт HTML4.

JavaScript (JS) — язык программирования, который поддерживает несколько парадигм программирования, которые представляют соответствующие стили программирования: объектно-ориентированный, императивный и функциональный. JavaScript обычно используется как язык сценариев для написания программ, интерпретируемых в браузерах для того, чтобы сделать веб-страницу интерактивной.

Основные практические работы курса базируются на стандарте HTML4 и каскадной таблице стилей CSS, а также использовании языка программирования JavaScript.

Главными достоинствами JavaScript являются легкость и наглядность программирования решения задачи, отображение математических выражений в том виде, в каком они обычно записываются на листе бумаги, простота использования, возможность создания средствами высококачественных технических возможностей интерактивных web-страниц с текстом, таблицами, изображениями, в том числе с псевдодинамикой.

---

[1] HTML (англ. *HyperText Markup Language)* — язык гипертекстовой разметки.

[2] Первым общедоступным описанием HTML был документ «Теги HTML», впервые упомянутый в Интернете Тимом Бернерсом-Ли в конце 1991 года.

[3] CSS (англ. *Cascading Style Sheets* — каскадные таблицы стилей.

[4] Стандарт HTML4 опубликован 18 декабря 1997 года.

[5] Стандарт HTML5 опубликован 28 октября 2014 года.

РАЗДЕЛ I. Основы создания web-страниц на языке форматирования HTML и каскадных таблиц стилей CSS

Тема 1. Базовые методы создания web-страниц на языке HTML

Учебная цель: сформировать умения разработки сценариев простых программных кодов, интерпретации их в браузере для создания элементарных приложений.

Учебно-справочный материал

По определению *интегрированная среда разработки* (ИСП)[1] — комплекс программных средств, используемый программистами для разработки программного обеспечения (ПО).

Среды программирования (или как их еще называют, среды разработки) — это программы, в которых программисты пишут свои программы. Иными словами, среда программирования служит для разработки (написания) программ и обычно ориентируется на конкретный язык или несколько языков программирования.

В общем случае интегрированная среда разработки включает в себя:

- текстовый редактор,
- компилятор и/или интерпретатор,
- средства автоматизации сборки,
- отладчик.

В нашем случае в качестве интегрированной среды программирования выступают следующие программные средства, необходимые для разработки программы:

1) *редактор* текста – в простейшем варианте Блокнот или любой другой редактор текста с подсветкой синтаксиса языка HTML (для написания текста сценария, так называемого программного кода);
2) *интерпретатор,* который транслирует программу, написанную на языке HTML во внутреннее представление кода, удобное для синтаксического анализа кода и исполнение фрагментов кода или всей программы.

План выполнения первого html-задания

В качестве примеров простых сценариев Вам предоставляются примеры ***Задание-1.html*** и ***Задание-2.html.*** Для изучения тегов HTML – компактный учебник ***HTML-book*** с подробным объяснением использования атрибутов тегов.

Прежде всего отметим, что текстовый файл сценария (с расширением ***html*** или ***htm,*** важно выбрать для себя конкретное расширение и в дальнейшем пользоваться только им), который Вы собираетесь создавать, а также изображения (с расширением ***jpg*** или ***gif***) или аудио, медиа файлы, которые собираетесь использовать, для простоты должны лежать в одной папке (в общем случае можно прописывать путь расположения всех этих файлов).

HTML – язык регистронезависимый (то есть не различает строчные и прописные буквы) и нестрогий. Нестрогость означает, что при ошибке в логике структуры программы интерпретатор применяет логику, заложенную по умолчанию (например, если тег открыт, но не закрыт, а должен быть закрыт, интерпретатор закроет его там, где посчитает нужным).

Изначально обучение написания сценариев рекомендуется методом «по образцу».

Принципы построения тегов языка HTML состоят, во-первых, в структурности, т. е. большинство тегов имеют открывающий тег и закрывающий тег, во-вторых, в принципе вложенности (как «матрешки»), т. е.

```
<HTML>                          открывающий тег: начало программы
  <HEAD>                        открывающий тег заголовка программы
     <TITLE>текст</TITLE>
  </HEAD>                       закрывающий тег заголовка программы
  <BODY>                        открывающий тег тела программы
  </BODY>                       закрывающий тег тела программы
</HTML>                         закрывающий тег: конец программы
```

[1] Интегрированная среда разработки (англ. Integrated development environment — IDE) называется также единой средой разработки (ECP)

Рассмотрим простой пример сценария: ***Задание-1.html***

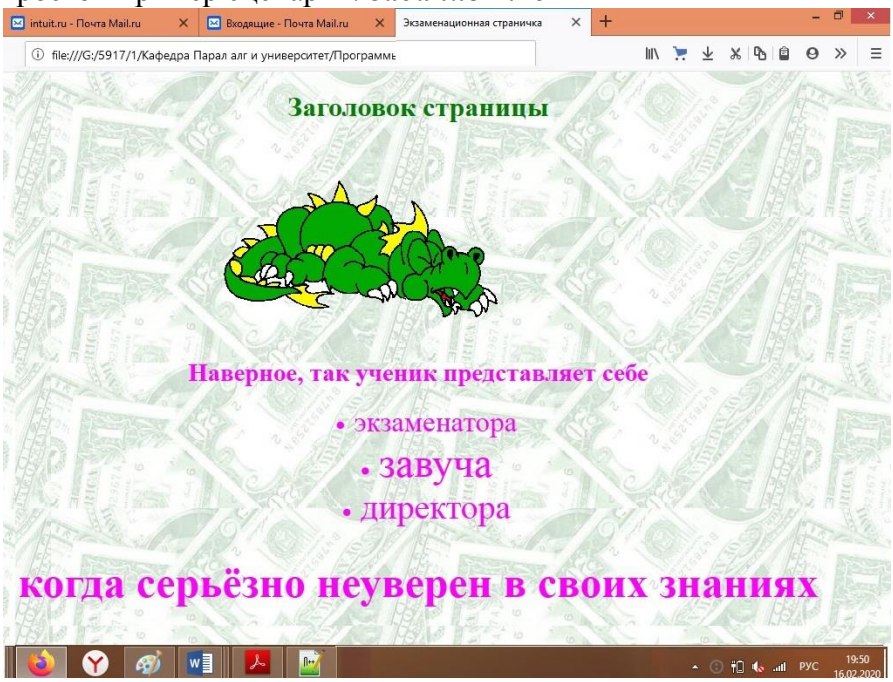


Рис. 1. Простой пример web-страницы ***Задание-1.html***

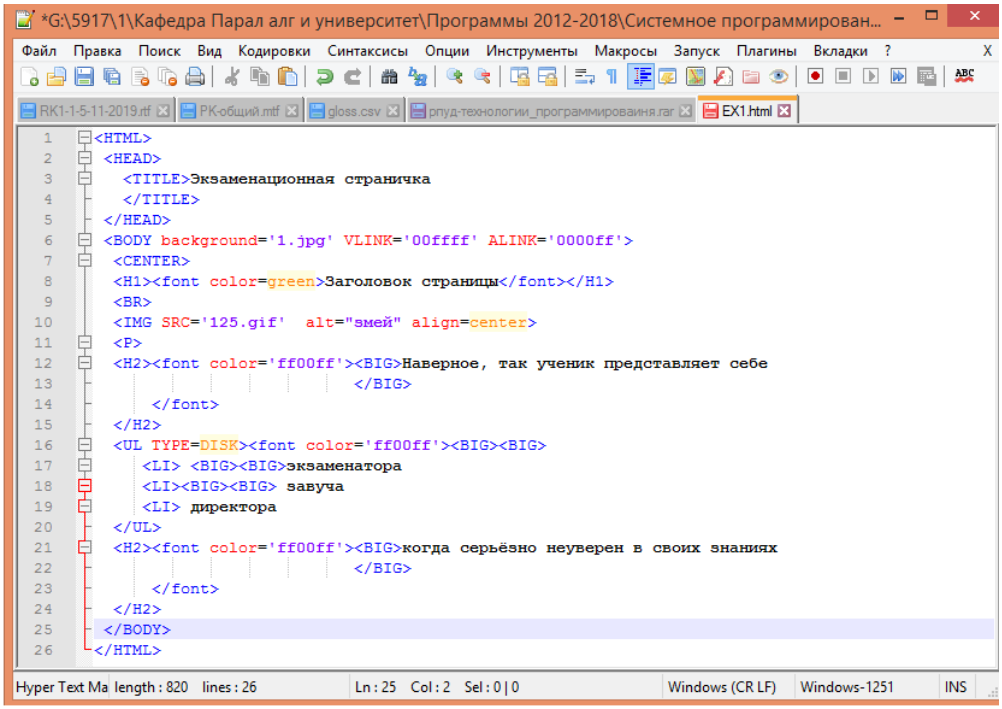


Рис. 2. Код сценария ***Задание-1.html***

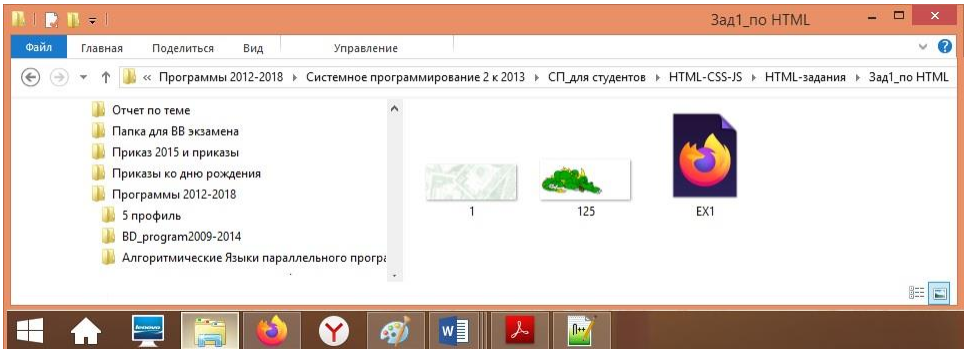


Рис.3. Папка с файлами: код сценария ***EX1.html***, двух изображений ***1.jpg*** (фона web-страницы) и ***125.jpg*** (изображения на web-странице)

Прежде чем написать свой сценарий web-страницы, скачайте папку ***Задание-1*** (рис. 3.) на свой компьютер и запустите на выполнение файл ***EX1.html.*** Откроется web-страница (рис. 1). код которой представлен на рисунке 2.

Внимательно проанализируйте код этого сценария и затем скопируйте этот код в специально созданную папку, творчески с ним поработайте, изменяя, добавляя теги с разными значениями атрибутов тегов (информацию о тегах и их атрибутах можно получить из справочника по языку HTML (html_book), предлагаемого в качестве материалов для выполнения практических работ по дисциплине «Системное программирование»), и сохраните с расширением htm или html.

Замечание 1. Полезно для себя выбрать конкретный тип расширения (htm/html) и в дальнейшем работать только с таким расширением, — это поможет избавиться от большинства ошибок, связанных с разными расширениями, и, главное, от потери времени на поиски такого рода ошибок.

Замечание 2. Обратите внимание, что для простоты изучения материала, но не нарушая общности, рекомендуется файлы изображений, видео и аудио помещать в ту же папку, в которой находится основной html-файл. В таком случае просто описывается путь доступа к вспомогательным файлам в виде *имя_файла.расширение*. Хотя стандартно принято web-страницу формировать из 2 объектов: собственно html-файла и папки со вспомогательными файлами (изображения, видео, аудио и, например, файлы с расширением css, js и прочее), которая должна иметь имя совпадающее с именем *html-файла* и расширение *files*.

Замечание 3. При создании web-страниц стандартно принято использовать изображения с расширением jpg/jpeg/gif. Другие расширения могут не отображаться у пользователей.

План выполнения html-заданий (папка HTML-задания)

Каждое из простых предлагаемых заданий предназначено для закрепления умений и навыков работы с определенными наборами тегов и атрибутами этих тегов (Рис. 4).

Рис. 4. Слева – изображение web-страницы, код которой представлен справа

Задание 1 поможет разобраться с возможностями размещения текста на странице, атрибутами шрифта, размещением и атрибутами изображений, заливкой фона и границ. При

этом код задания сформирован таким образом, чтобы наглядно продемонстрировать свойства атрибутов тегов.

Рекомендуется выполнить самостоятельно и творчески не менее шести вариантов, причем, включить в выполнение такой из вариантов, когда два рисунка расположены на одном уровне справа и слева на странице.

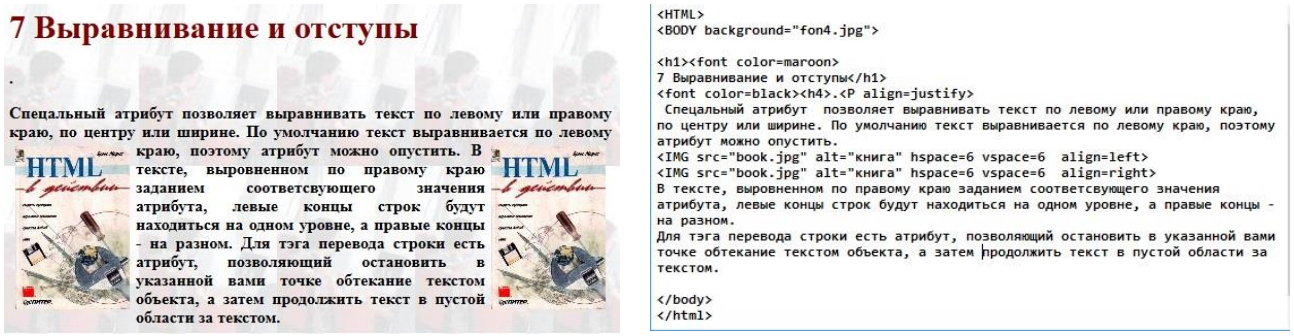


Рис. 5. Расположение на web-странице рисунков на одном уровне

Задание 2 предполагает изучение возможностей вставки ссылок внешней и внутренней (между страницами сайта), а также организации бугущей строки на web-странице. Бегущая строка (движущийся дед-Мороз на фоне звездного неба) позволяет оживить web-страницу, организуется тегом ***<marquee>***.

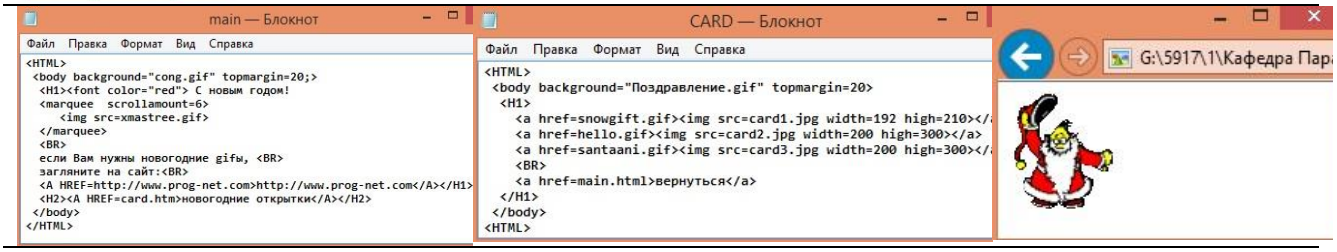


Рис. 6.Изображение и код основной web-страницы

Рис. 7. Изображение и код web-страницы, полученной при переходе по внутренней ссылке «новогодние открытки»

Рис. 8. Изображение web-страницы, полученной при переходе по ссылке, представляющей открытку

Со страницы, представляющей новогоднюю открытку можно вернуться назад, только нажав на стрелку возврата на предыдущую web-страницу (стрелка вверху слева). На рисунке 7 представлен текст, в котором присутствует ссылка возврата на основную web-страницу, организованная разработчиком web-страницы.

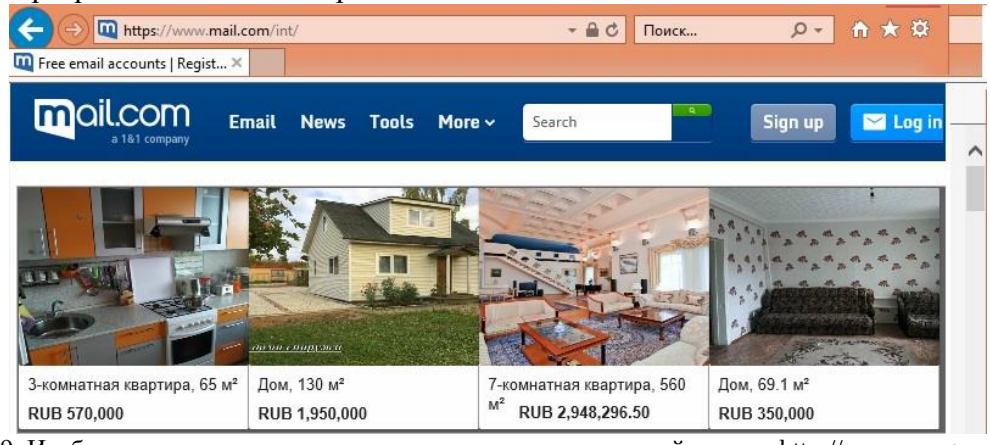


Рис. 9. Изображение страницы, на которую перешли по внешней ссылке http://www.prog-net.com

Задание 3 предполагает изучение возможностей организации таблиц на web-странице или организации web-страниц с помощью таблиц. Кроме этого, с помощью таблиц можно выполнять построение схем и чертежей, что значительно уменьшает объём электронного продукта.
Для построения таблиц, прежде всего, используются теги ***<TABLE>***, ***<TR>***, ***<TD>***, ***<DD>***.Таблицы могут быть с объединением ячеек по строкам, по столбцам, а также более сложным образом. Простейший вариант предлагается в Задании 3, более сложные варианты желательно проработать самостоятельно, используя материалы html_book.

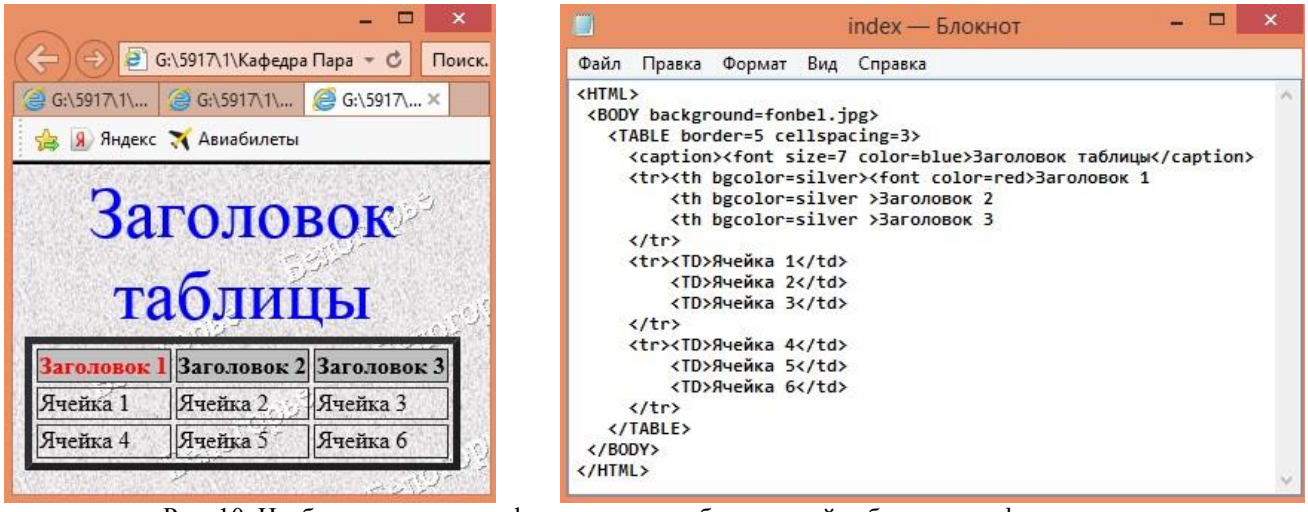
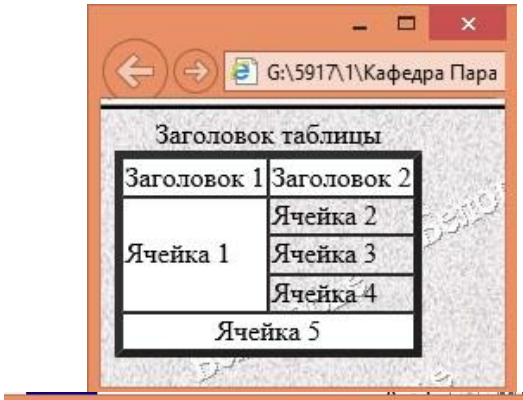


Рис. 10. Изображение и код web-страницы, отображающий таблицу на web-странице



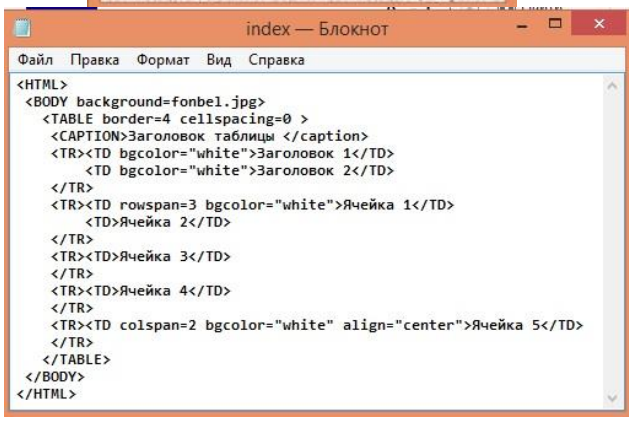
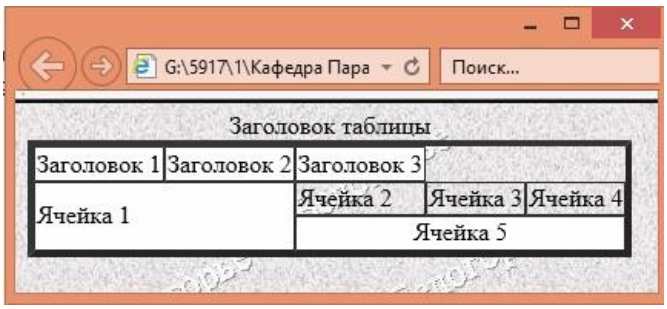


Рис. 11. Пример объединения ячеек отдельно по строкам или по столбцам (вид web-страницы и код)



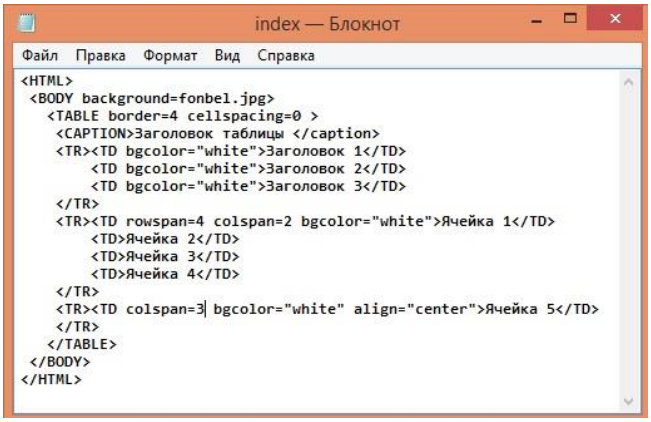


Рис. 12. Пример объединения ячеек одновременно по строкам и столбцам (вид web-страницы и код)

На рисунке 11 и 12 представлены изображения web-страниц и коды к ним с организацией объединения строк или столбцов отдельно и в одновременном использовании в одной таблице.

На рисунке 11 «Ячейка 1» представляет объединение трёх ячеек в одну; этот результат достигается применением атрибута ***rowspan*** со значением ***3*** в теге ***<TD>***: ***<TD rowspan=3>.*** Чтобы получить объединение двух ячеек в строке («Ячейка 5») в теге ***<TD>*** используется атрибут ***colspan***: ***<TD colspan=2>.***

На рисунке 12 используется объединение трёх ячеек в строке («Ячейка 5») с помощью атрибута ***colspan*** аналогично приёму, описанному в примере 11. Одновременное объединение ячеек по строке и по столбца демонстрируется при организации «Ячейки 1» одновременным применением атрибутов ***rowspan*** и ***colspan***: ***<TD rowspan=2 colspan=2>***

Замечание. Обратите внимание, что таблица в HTML всегда прямоугольная: количество ячеек в каждой строке таблицы равно числу ячеек в строке таблицы с максимально определенным количеством ячеек. Таблица, представленная на рисунке 12 содержит ошибку построения: в первой строке описаны 4 ячейки, а максимально в строке должно быть 5 ячеек. Поэтому информацию можно записать лишь в первые четыре ячейки (непосредственно описанные), а в пятую ячейку записать ничего нельзя.

Ниже на рисунке 13 представлена организация вложенности одной таблицы в другую (ячейка 2 содержит таблицу размерностью 3 x 2).

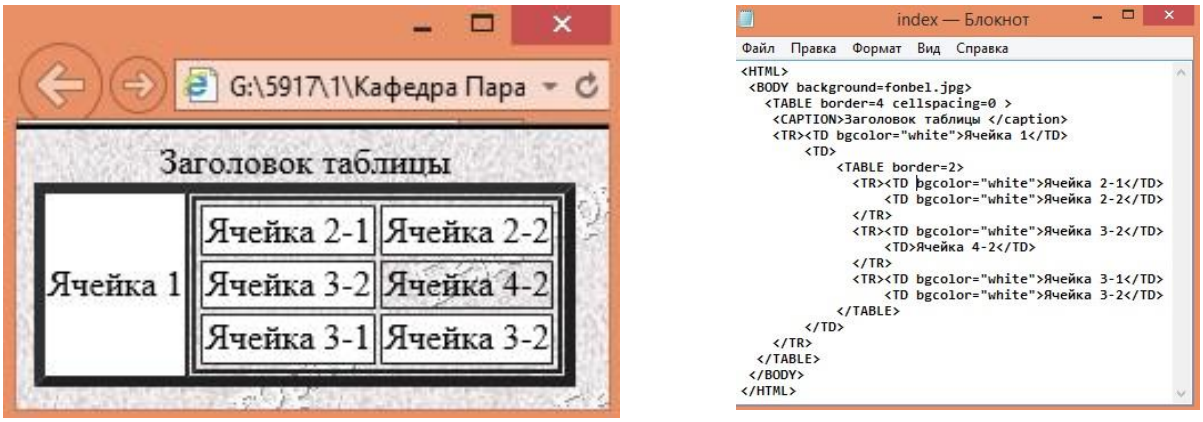


Рис. 13. Изображение и код web-страницы, содержащей две таблицы, одна из которых вложена в другую

Рассмотрим теперь пример неявного применения таблицы (рис. 14).

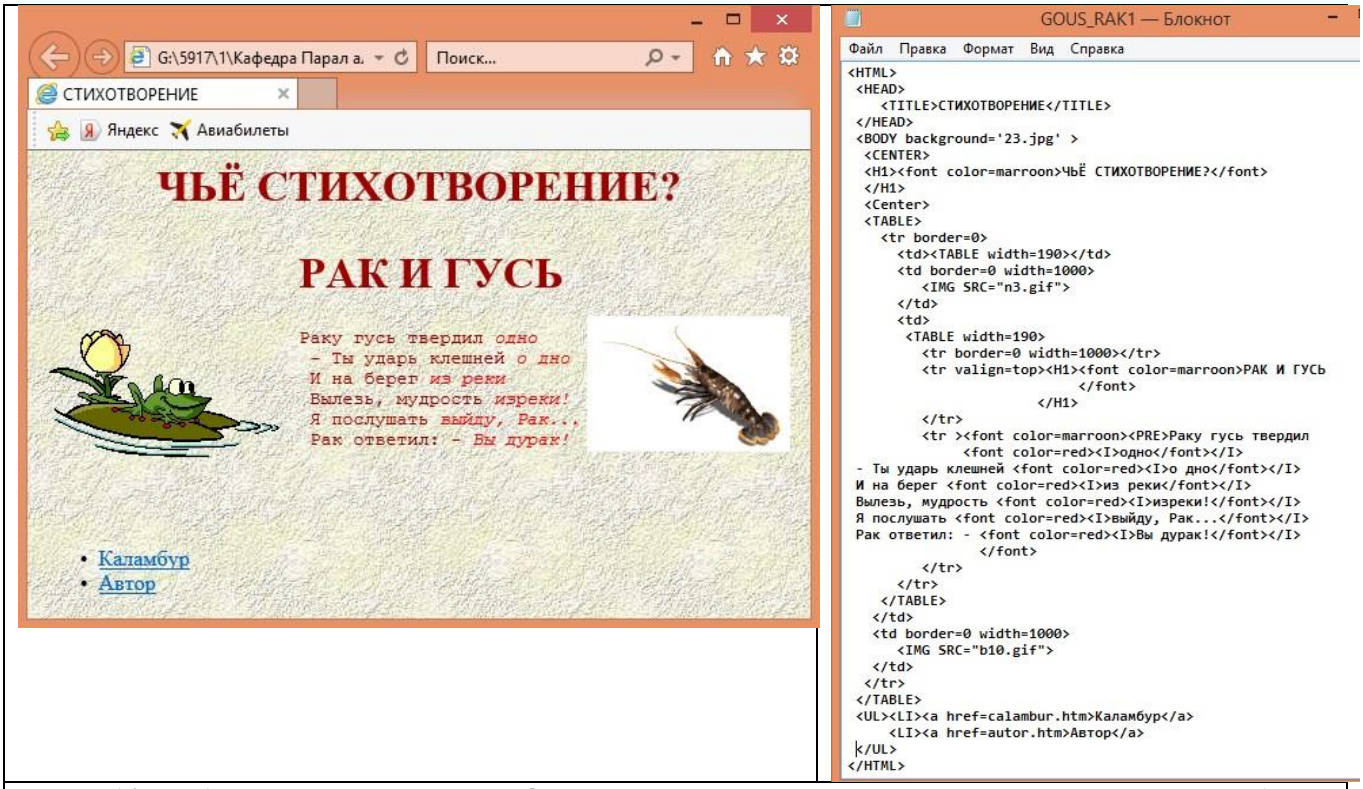
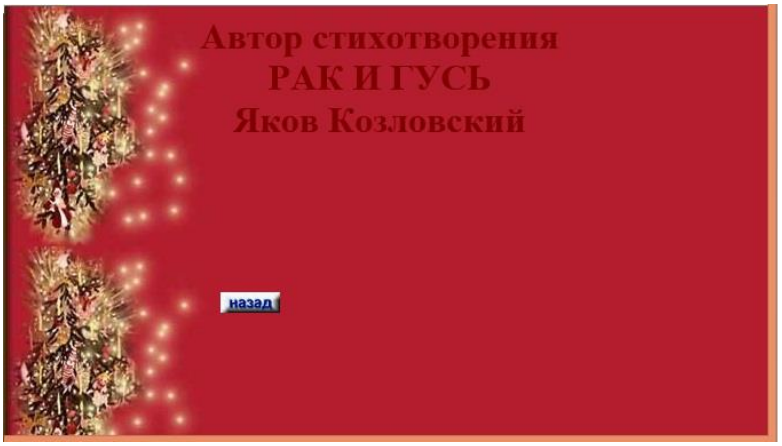


Рис. 14. Изображение и код основной web-страницы, иллюстрирующий известное стихотворение-каламбур

Ниже представлены изображения web-страниц, на которые делается переход по ссылкам “Автор” (рис. 15) и “Каламбур” (рис. 16) и их коды. На рисунках 15 и 16 изображение кнопки “назад” является ссылкой на основную web-страницу (рис. 14).

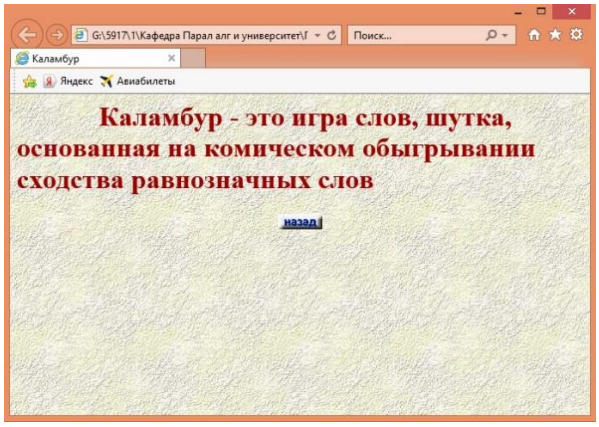


AUTOR — Блокнот

Файл Правка Формат Вид Справка

```
<HTML>
 <HEAD>
  <TITLE>autor</TITLE>
 </HEAD>
 <BODY background='6.jpg'>
   <CENTER>
   <H1><font color=maroon align=right>
     <B>Автор стихотворения <BR>
        РАК И ГУСЬ<BR>
        Яков Козловский
     </B>
    </font>
 </H1>
 </CENTER><BR><BR><BR><BR><BR>
            
            
            
            

  <a href=gous_rak.htm> <IMG SRC="but_bac.gif"></a>
</BODY>
</HTML>
```

Рис. 15. Изображение web-страницы, появляющейся при переходе по ссылке “Автор”, и её код



CALAMBUR1 — Блокнот

Файл Правка Формат Вид Справка

```
<HTML>
 <HEAD>
  <TITLE>Каламбур</TITLE>
 </HEAD>
 <BODY background='23.jpg'>
   <H1><font color=marroon>
              
Каламбур - это игра слов, шутка, основанная
на комическом обыгрывании сходства равнозначных слов
     </font>
   </H1>
   <CENTER>
    <a href=gous_rak.htm><IMG SRC="but_bac.gif"></a>
   </CENTER>
 </BODY>
</HTML>
```

Рис. 16. Изображение web-страницы, появляющейся при переходе по ссылке “Каламбур”, и её код

Теперь рассмотрим Задание 7, которое тематически ближе к рассмотренным. Смысл задания состоит в том, чтобы организовать web-страницу так, чтобы обеспечить быстрый доступ к определенным разделам длинного текста (например, статьи). Принцип организации такого доступа состоит в том, что элементы оглавления, расположенные в начале такого текста являются ссылками на заранее отмеченные участки этого текста. В нашем случае отмеченными являются позиции перед фрагментами «Статья 272», «Статья 273», «Статья 274», а также «ГЛАВА 28». Отметка позиции выполняется специальным видом ссылки-отметки ***<a name=начало>***, где слово ***начало***, на самом деле, может быть любой допустимой цепочкой символов (идентификатор). Ссылка-отметка позиции, например, выглядит следующим образом: ГЛАВА 28. ПРЕСТУПЛЕНИЯ В СФЕРЕ КОМПЬЮТЕРНОЙ ИНФОРМАЦИИ.

ГЛАВА 28. ПРЕСТУПЛЕНИЯ В СФЕРЕ КОМПЬЮТЕРНОЙ ИНФОРМАЦИИ

Статья 272 Статья 273 Статья 274

Статья 272. Неправомерный доступ к компьютерной информации

1. Неправомерный доступ к охраняемой законом компьютерной информации, то есть информации на машинном носителе, в электронно-вычислительной машине (ЭВМ), системе ЭВМ или их сети, если это деяние повлекло уничтожение, блокирование, модификацию либо копирование информации, нарушение работы ЭВМ, системы ЭВМ или их сети, - наказывается штрафом в размере от двухсот до пятисот минимальных размеров оплаты труда или в размере заработной платы или иного дохода осужденного за период от двух до пяти месяцев, либо исправительными работами на срок от шести месяцев до одного года, либо лишением свободы на срок до двух лет.

2. То же деяние, совершенное группой лиц по предварительному сговору или организованной группой либо лицом с использованием своего служебного положения, а равно имеющим доступ к ЭВМ, системе ЭВМ или их сети, - наказывается штрафом в размере от пятисот до восьмисот минимальных размеров оплаты труда или в размере заработной платы или иного дохода осужденного за период от пяти до восьми месяцев, либо исправительными работами на срок от одного года до двух лет, либо арестом на срок от трех до шести месяцев, либо лишением свободы на срок до пяти лет.

к началу...

Статья 273. Создание, использование и распространение вредоносных программ для ЭВМ

1. Создание программ для ЭВМ или внесение изменений в существующие программы, заведомо приводящих к несанкционированному уничтожению, блокированию, модификации либо копированию информации, нарушению работы ЭВМ, системы ЭВМ или их сети, а равно использование либо распространение таких программ или машинных носителей с такими программами - наказываются лишением свободы на срок до трех лет со штрафом в размере от двухсот до пятисот минимальных размеров оплаты труда или в размере заработной платы или иного дохода осужденного за период от двух до пяти месяцев.

2. Те же деяния, повлекшие по неосторожности тяжкие последствия, - наказываются лишением свободы на срок от трех до семи лет.

к началу...

Статья 274. Нарушение правил эксплуатации ЭВМ, системы ЭВМ или их сети

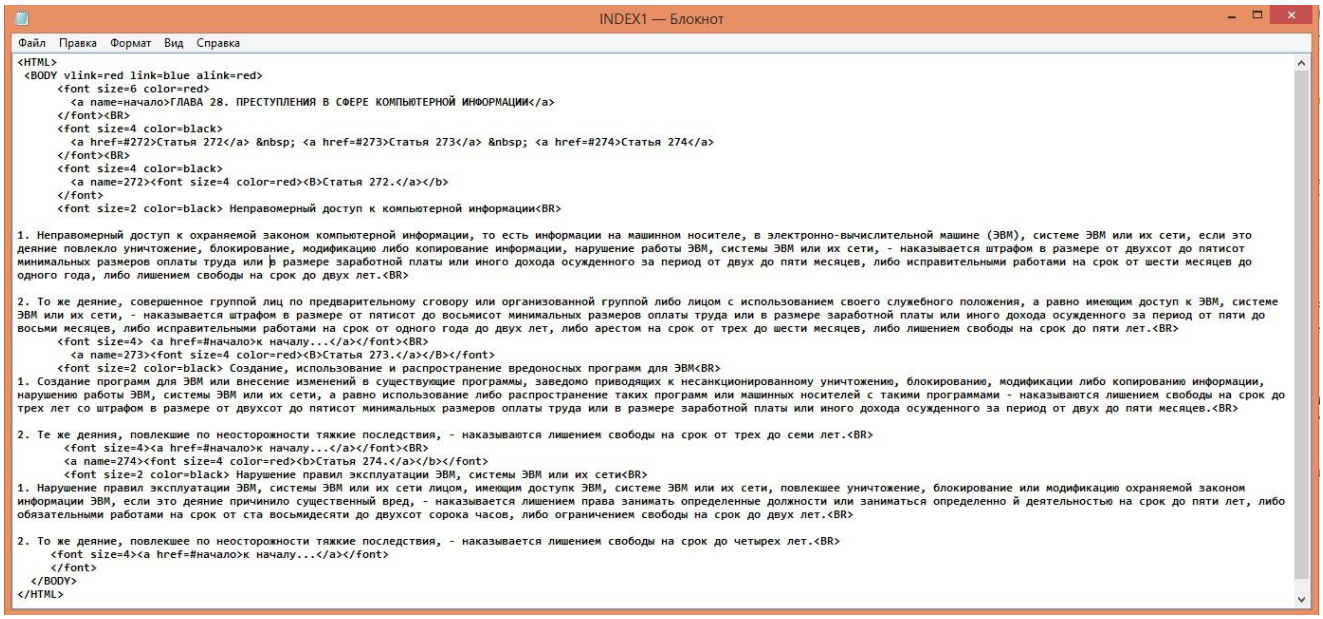
1. Нарушение правил эксплуатации ЭВМ, системы ЭВМ или их сети лицом, имеющим доступ ЭВМ, системе ЭВМ или их сети, повлекшее уничтожение, блокирование или модификацию охраняемой законом информации ЭВМ, если это деяние причинило существенный вред, - наказывается лишением права занимать определенные должности или заниматься определенно й деятельностью на срок до пяти лет, либо обязательными работами на срок от ста восьмидесяти до двухсот сорока часов, либо ограничением свободы на срок до двух лет.

2. То же деяние, повлекшее по неосторожности тяжкие последствия, - наказывается лишением свободы на срок до четырех лет.

к началу...

Рис. 17. Изображение web-страницы, представляющей длинный текст со ссылками на фрагменты

Приведенный перед рисунком 17 фрагмент кода с идентификатором «начало» отмечает позицию перед словом «ГЛАВА».

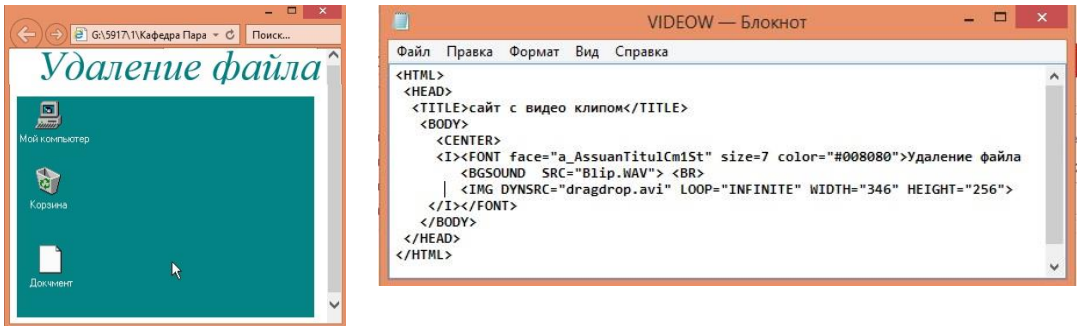



```
<HTML>
 <BODY vlink=red link=blue alink=red>
      <font size=6 color=red>
        <a name=начало>ГЛАВА 28. ПРЕСТУПЛЕНИЯ В СФЕРЕ КОМПЬЮТЕРНОЙ ИНФОРМАЦИИ</a>
      </font><BR>
      <font size=4 color=black>
        <a href=#272>Статья 272</a>   <a href=#273>Статья 273</a>   <a href=#274>Статья 274</a>
      </font><BR>
      <font size=4 color=black>
        <a name=272><font size=4 color=red><B>Статья 272.</a></b>
      </font>
      <font size=2 color=black> Неправомерный доступ к компьютерной информации<BR>

1. Неправомерный доступ к охраняемой законом компьютерной информации, то есть информации на машинном носителе, в электронно-вычислительной машине (ЭВМ), системе ЭВМ или их сети, если это
деяние повлекло уничтожение, блокирование, модификацию либо копирование информации, нарушение работы ЭВМ, системы ЭВМ или их сети, - наказывается штрафом в размере от двухсот до пятисот
минимальных размеров оплаты труда или в размере заработной платы или иного дохода осужденного за период от двух до пяти месяцев, либо исправительными работами на срок от шести месяцев до
одного года, либо лишением свободы на срок до двух лет.<BR>

2. То же деяние, совершенное группой лиц по предварительному сговору или организованной группой либо лицом с использованием своего служебного положения, а равно имеющим доступ к ЭВМ, системе
ЭВМ или их сети, - наказывается штрафом в размере от пятисот до восьмисот минимальных размеров оплаты труда или в размере заработной платы или иного дохода осужденного за период от пяти до
восьми месяцев, либо исправительными работами на срок от одного года до двух лет, либо арестом на срок от трех до шести месяцев, либо лишением свободы на срок до пяти лет.<BR>
      <font size=4> <a href=#начало>к началу...</a></font><BR>
        <a name=273><font size=4 color=red><B>Статья 273.</a></B></font>
      <font size=2 color=black> Создание, использование и распространение вредоносных программ для ЭВМ<BR>
1. Создание программ для ЭВМ или внесение изменений в существующие программы, заведомо приводящих к несанкционированному уничтожению, блокированию, модификации либо копированию информации,
нарушению работы ЭВМ, системы ЭВМ или их сети, а равно использование либо распространение таких программ или машинных носителей с такими программами - наказываются лишением свободы на срок до
трех лет со штрафом в размере от двухсот до пятисот минимальных размеров оплаты труда или в размере заработной платы или иного дохода осужденного за период от двух до пяти месяцев.<BR>

2. Те же деяния, повлекшие по неосторожности тяжкие последствия, - наказываются лишением свободы на срок от трех до семи лет.<BR>
       <font size=4><a href=#начало>к началу...</a></font><BR>
       <a name=274><font size=4 color=red><b>Статья 274.</a></b></font>
       <font size=2 color=black> Нарушение правил эксплуатации ЭВМ, системы ЭВМ или их сети<BR>
1. Нарушение правил эксплуатации ЭВМ, системы ЭВМ или их сети лицом, имеющим доступк ЭВМ, системе ЭВМ или их сети, повлекшее уничтожение, блокирование или модификацию охраняемой законом
информации ЭВМ, если это деяние причинило существенный вред, - наказывается лишением права занимать определенные должности или заниматься определенно й деятельностью на срок до пяти лет, либо
обязательными работами на срок от ста восьмидесяти до двухсот сорока часов, либо ограничением свободы на срок до двух лет.<BR>

2. То же деяние, повлекшее по неосторожности тяжкие последствия, - наказывается лишением свободы на срок до четырех лет.<BR>
     <font size=4><a href=#начало>к началу...</a></font>
     </font>
  </BODY>
</HTML>
```

Рис. 18. Код web-страницы, изображенной на рисунке 17.

Для обращения к отмеченным позициям используются, так называемые, “внутренние ссылки”. Так, чтобы обратиться из некоторого места текста к началу текста в примере используется “внутренняя” ссылка ***к началу...***, где «***начало»*** — тот идентификатор, который был использован в ссылке-отметке позиции. В отличие от ссылки на другую web-страницу, в которой прописывается путь, куда требуется перейти, во “внутренней” ссылке используется идентификатор, прописанный в ссылке-отметке позиции, перед которым ставится знак “#”.

Задание 5 являет собой простейший пример добавления на web-странице аудио и видео файлов. На сегодняшний день существует большое разнообразие расширений для таких файлов и разнообразие способов подключений их к web-странице, поэтому это задание следует рассматривать как первое знакомство с методикой подключения таких файлов к web-странице.




```
<HTML>
 <HEAD>
  <TITLE>сайт с видео клипом</TITLE>
   <BODY>
     <CENTER>
     <I><FONT face="a_AssuanTitulCm1St" size=7 color="#008080">Удаление файла
        <BGSOUND  SRC="Blip.WAV"> <BR>
       | <IMG DYNSRC="dragdrop.avi" LOOP="INFINITE" WIDTH="346" HEIGHT="256">
    </I></FONT>
   </BODY>
 </HEAD>
</HTML>
```

Рис. 19. Изображение web-страницы, включающей видео и аудио файлы, и её код

К сожалению, в этой методичке невозможно показать ни движение объектов, расположенных на web-странице, ни услышать звуки, сопровождающие движение.

В этом примере звук добавляется на web-страницу тегом ***<BGSOUND>*** с обязательным атрибутом ***SRC***, значением которого является путь к звуковому файлу, а именно: ***<BGSOUND SRC="Blip.WAV">***. Видео добавляется с помощью уже известного тега ***<IMG>***, с обязательным атрибутом ***DYNSRC***, значением которого как и в случае звукового файла является путь к видео файлу, а именно: ***<IMG DYNSRC="dragdrop.avi">.*** В примере 19 это строка ***<IMG DYNSRC="dragdrop.avi" LOOP="INFINITE" WIDTH="346" HEIGHT="256">.***
Оставшиеся два задания Задание 4 и Задание 6 касаются использования фреймов. На сегодняшний день фреймы используются не часто, но интересна методика использования их для организации карты ссылок (Задание 6). Но для того, чтобы был понятен процесс организации

карты ссылок необходимо познакомиться с простой организацией нескольких фреймов на web-странице.

Задание 4 дает понятие об организации сайта на основе фреймов. Важно заметить, что методика организации такого сайта требует n+1 полноценной web-страницы, так как одна web-страница представляет схему расположения основных и дополнительных web-страниц.

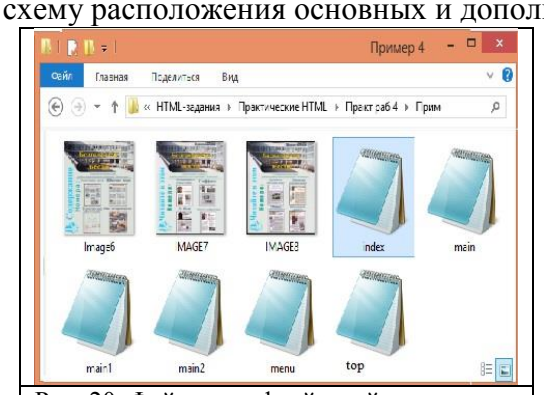


Рис. 20. Файлы для фрейм-сайта

Основная web-страница – структура – html-файл index.html. Структура состоит из трёх областей: изначально область html-страницы делится на две вертикальные области тегом <***FRAMESET***> и атрибутом в нем ***COLS="200,*"*** (левая область – 200 пикселей, правая – остальная область). Левую область по умолчанию заполняем web-страницей ***menu.html*** (которая содержит ссылки, для замены содержимого в горизонтальной области, о которой скажем в следующем абзаце), используя тег <***FRAME***>: <***FRAME SRC="menu.html"***> (рисунки 21 и 22).

Правую область также делим на две горизонтальные части: верхнюю для заголовка с бегущей строкой и нижнюю – для фреймов с основным содержанием. Деление области осуществляем тегом <***FRAMESET***> с атрибутом ***ROWS="70,*".*** Отметим, что для этой области это уже вложенный тег <***FRAMESET***>. Для заполнения верхней области используем <***FRAME***>: <***FRAME SRC="top.html"***>, а для нижней — <***FRAME SRC="main.html" NAME="OKNO"***> с файлом ***main.html*** по умолчанию.

Рис. 21. Изображение основной web-страницы (структура, наполненная фреймами)

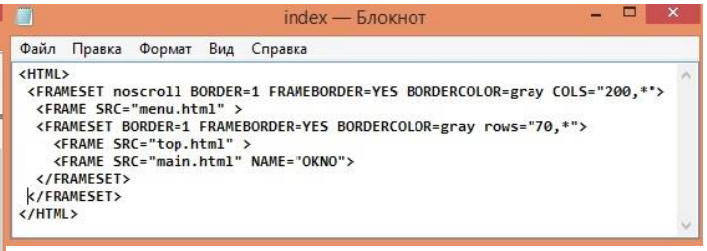


Рис. 22. Код web-документа, представляющего структуру

Теперь покажем состав остальных web-страниц, входящих в состав фрейм-сайта. Назовем левую область основной web-страницы web-страницей меню ссылок, верхнюю область правой части топовой web-страницей фреймого сайта и нижнюю часть правой области фрейм-сайта web-страницей содержимого.

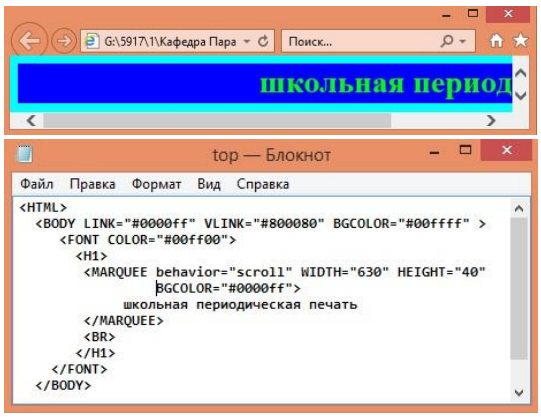


Рис. 23. Изображение и код топовой web-страницы с бегущей строкой

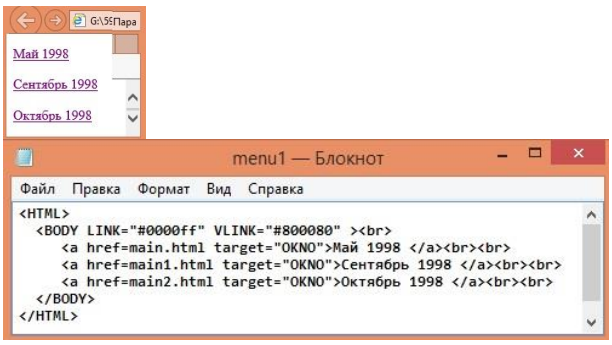


Рис. 24. Изображение код web-страницы, представляющей меню выбора содержимого правой чати фрейм-сайта

Рис. 25. Изображения и коды web-страниц содержания фрейм-сайта

Наконец, рассмотрим последнее Задание 6, которое иллюстрирует представление изображение в виде карты ссылок.

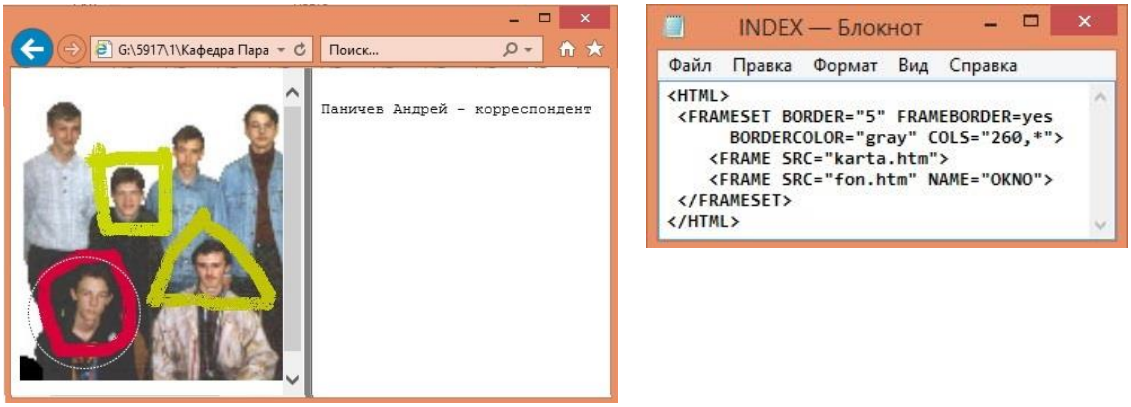


Рис. 26. Изображение, представляющее карту, элементы которой являются ссылками

В этом примере применяется та же организация фреймовой web-страницы. Структура состоит из двух областей, полученных разделением с помощью атрибута ***COLS***. В левую область поместим web-страницу с отмеченными областями разной формы, где каждая область представляет ссылку.

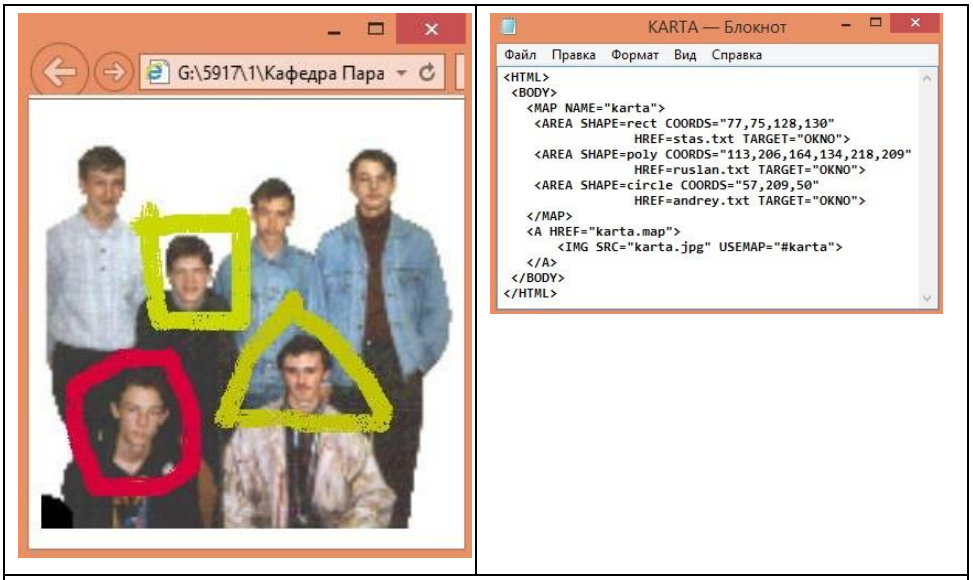


Рис. 27. Изображение и код карты ссылок с разметкой областей

Методика организации карты ссылок состоит в применении спецциального тега ***<MAP>*** с атрибутом, представляющим идентификатор этой карты: ***<MAP NAME="karta">***, указанием к какому изображению будет применена эта карта ***<IMG SRC="karta.jpg" USEMAP="#karta">***, отметкой областей, которые впоследствии станут ссылками и обязательным атрибутом ***USEMAP***, который обеспечивает преобразование отмеченных областей в ссылки.

Отмечаются области в рамках карты ***<MAP>***. Отмечать можно области разной формы: прямоугольник, треугольник, окружность и область произвольной формы:

<AREA SHAPE=rect COORDS="77,75,128,130" HREF=stas.txt TARGET="OKNO">
<AREA SHAPE=poly COORDS="113,206,164,134,218,209" HREF=ruslan.txt TARGET="OKNO">
<AREA SHAPE=circle COORDS="57,209,50" HREF=andrey.txt TARGET="OKNO">

Координаты точек этих областей можно определить с помощью различных графических программ, например, ***PAINT***, ***PHOTOSHOP***.

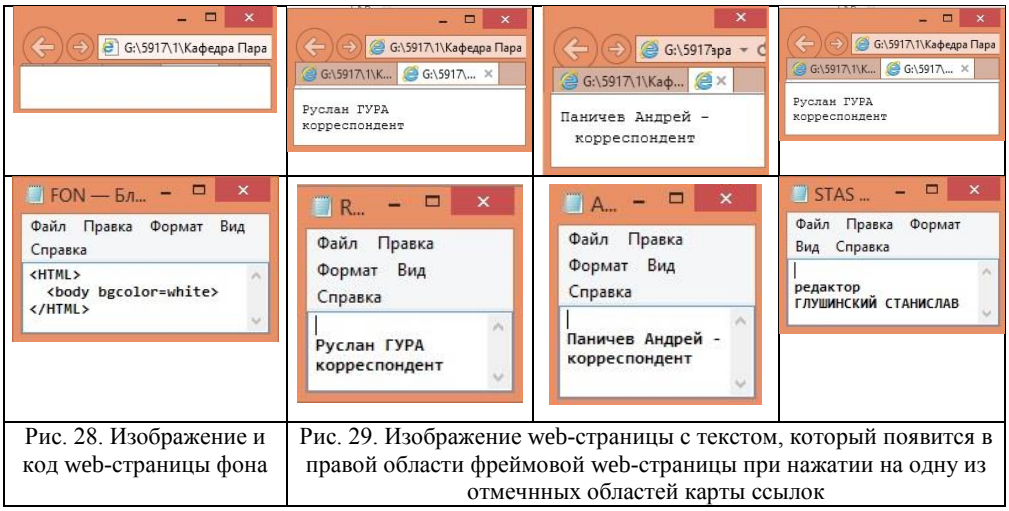


Рис. 28. Изображение и код web-страницы фона

Рис. 29. Изображение web-страницы с текстом, который появится в правой области фреймовой web-страницы при нажатии на одну из отмечнных областей карты ссылок

На основе рассмотренных базовых методов организации web-документов создайте свои творческие примеры, в которых бы были использованы изученные теги и приёмы.

Тема 2. Основные приемы организации web-страниц с использованием каскадной таблицы стилей CSS

Каскадная таблица стилей (CSS) является составной частью стандарта HTML4 и предназначена для ювелирного форматирования web-страницы. Аналогично атрибутам тегов HTML в каскадных таблицах стилей рассматриваются селекторы. Получить подробную информацию о селекторах CSS можно, например, в предлагаемом в учебных материалах по изучению дисциплины «Системное программирование» электронном учебнике CSS_book.

Задания, предлагаемые в настоящем методическом пособии помогут получить более современные умения и навыки формирования web-страниц.

В истории организации web-сайтов был период, когда конкурировали методики организации web-страниц с помощью фреймов и таблиц. Каждая из этих методик имеет свои особенности. На сегодняшний день фреймовые web-сайты практически не используются, применяют тег <***IFRAME***> как элемент web-страницы – «окошко» с прокруткой, конкурируют методики создания web-страниц с помощью таблиц и блоков. Блоки (или блочная разметка) формируются с помощью <***DIV***> тегов. В CSS крме блочной разметки используется и строковая разметка с помощью тега <***SPAN***>.

Познакомимся с использованием <***DIV***> тега и селекторами CSS на примере заданий этого методического пособия. Кроме этого настоятельно рекомендуется выполнить задания из электронного пособия ***css.pdf***, предлагаемого в методических материалах для выполнения практических работ по дисциплине «Системное программирование».

Блок, определяемый тегом <***DIV***>, используется для группировки различных селекторов для удобства использования как для одной web-страницы, так и для нескольких в случае внешнего размещения на отдельной web-странице и подключения в дальнейшем к нужным web-страницам. Блоки можно вкладывать друг в друга, но пересекаться блоки не должны.

Строковая разметка <***SPAN***> не приводитк образованию блока текста, с помощью <***SPAN***> заменяют текстовые теги <***I***>, <***B***>, <***FONT***>, <***U***>, <***SUB***>, <***SUP***>, а блоковая разметка универсальна, она включает не только формирование внутренней структуры блока, но и размещение блока относительно страницы или объемлющих элементов.

Пример описания стиля блока, описываемого тегом <***DIV***>:

<DIV style="height:50px; width:200px; margin-left:15px; padding-left:10px; border-width:2px; border-color:#0000ff; border-style:groove; float:right"> текст текст текст текст текст текст текст текст </DIV>

Замечание. Селекторы разделяются точкой с запятой, поэтому последний селектор не заканчивается точкой с запятой.

Стиль этого блока описан следующими селекторами:

height:50px; — описывается высота блока в 50 пикселей;

width:200px; — описывается ширина блока в 200 пикселей;

margin-left:15px; —отступ слева от границы страницы в 15 пикселей;

padding-left:10px; —отступ слева от границы блока до текста – 10 пикселей;

border-width:2px; ширина границы блока — 2 пикселя;

border-color:#0000ff; —цвет границы – синий (***#0000ff***);

border-style:groove; —вид границы;

float:right — расположение блока относительно страницы — справа (или относительно другого блока, который расположен следующим справа).

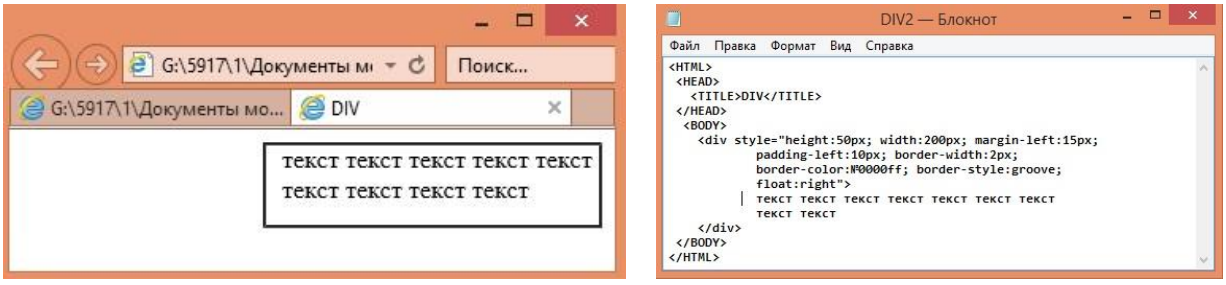


Рис. 30. Изображение и код web-страницы, созданной с помощью блочной разметки

Покажем теперь каким образом выполнить более сложные построения с помощью блочной разметки.

Задание 1. Сформируем четыре блока по два в каждой горизонтали. Для каждого блока определим разные границы: разного типа, разного цвета, разной толщины, определим разную заливку и различные другие характеристики.

Обратите внимание, что четвёртый блок (справа внизу) организован с невидимой границей. Внутри этого блока расположен текст разного типа: курсив, жирный, увеличенный жирный…

Обратите внимание, в каком порядке описаны блоки и как они располагаются на web-странице.

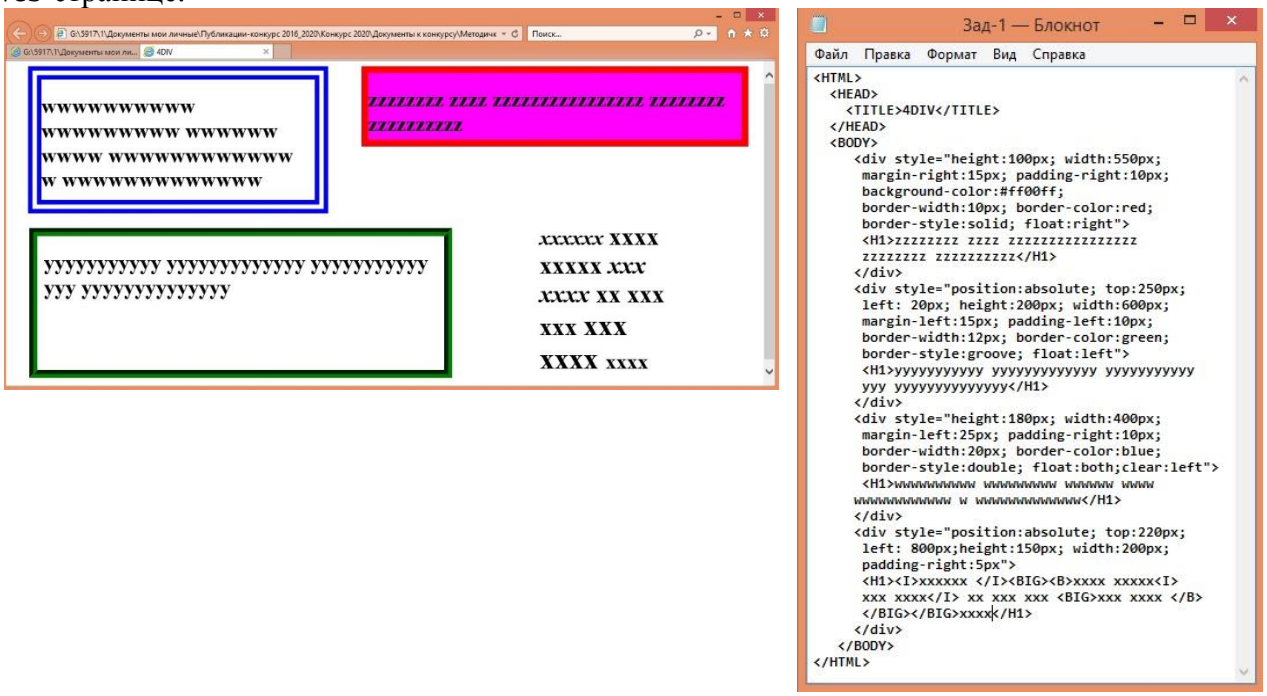


Рис. 31. Изображение web-страницы с четырьмя блоками и код этой web-страницы

Блоки могут быть вложенными. Попытайтесь выполнить формирование web-страницы, в которой внутри одного блока располагаются другие блоки с разным наполнением.

Описание блока или блоков может быть расположено вне основной web-страницы в виде внешнего блока и подключаться к требуемым web-страницам.

Задание 2. Реализуйте внешнее представление блочной разметки. Кроме этого, реализуйте блок с текстом, выровненным по левому краю, с отступом первой строки и буквицей, а также разрядкой текста на 3 пункта. Добавьте фоновый рисунок на web-странице и вставьте математическую формулу с греческими символами и нижними или верхними индексами.

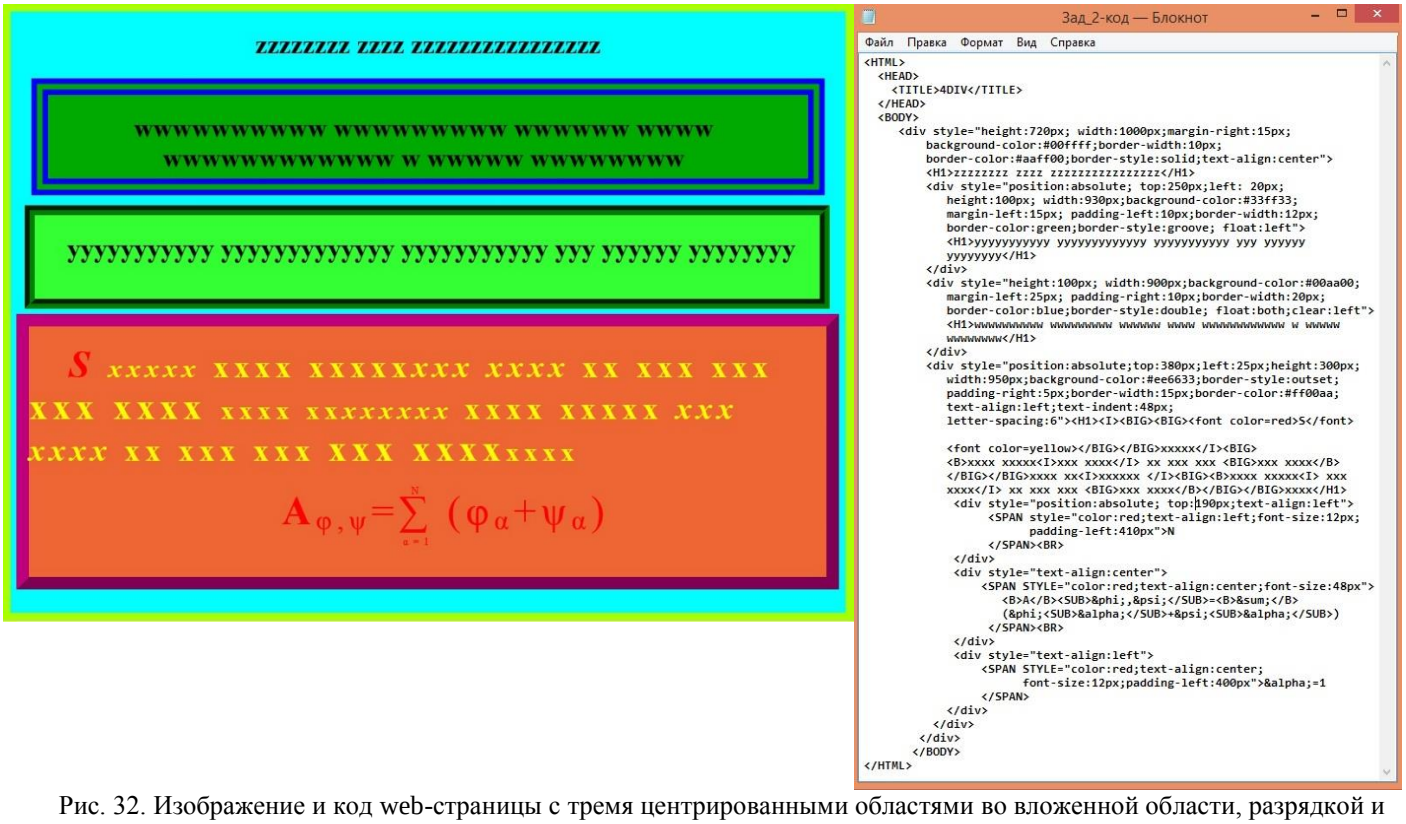


Рис. 32. Изображение и код web-страницы с тремя центрированными областями во вложенной области, разрядкой и буквицей в третьей вложенной области, а также формулой

Проанализируйте структуру вложенности блоков <***DIV***>.

Размеры объемлющего блока должны учитывать размеры внутреннего блока, но и размеры отступов справа и слева от текста и от страницы (если они есть), и размеры толщины границ. Глубина вложенности может быть любой, лишь бы правильно были записаны открывающие и закрывающие теги <***DIV***>.

Каждый из вложенных блоков–потомков объемлющего блока имеют свою заливку фона, определяемую селектором ***background-color:***значение (представляемое названием цвета или кодом RGB), например, ***background-color:#00ffff;***, толщину, вид и цвет границы блока, например, ***border-color:#aaff00;border-style:solid;*** размещение текста в блоке и другие параметры.

Значения селекторов, указанных в объемлющем блоке, являются значениями по умолчанию для параметров вложенных блоков, если эти селекторы не присутствуют в описании разметки вложенного блока. Если же во вложенном блоке присутствуют селекторы, одинаковые с селекторами объемлющего блока, то приоритетными являются значения соответствующих селекторов вложенного блока.

Таким образом в примере размечены первый и второй вложенные блоки (с центрированным текстом). Третий вложенный блок состоит из собственно своего блока и трех вложенных блоков второго уровня, формирующих формулу. Собственно третий блок, во-первых, формирует текст, выровненный по левому краю и разреженный на 6 пунктов, во-вторых, имеет отступ первой строки и буквицу. В формировании формулы участвует тег строковой разметки <***SPAN***>.

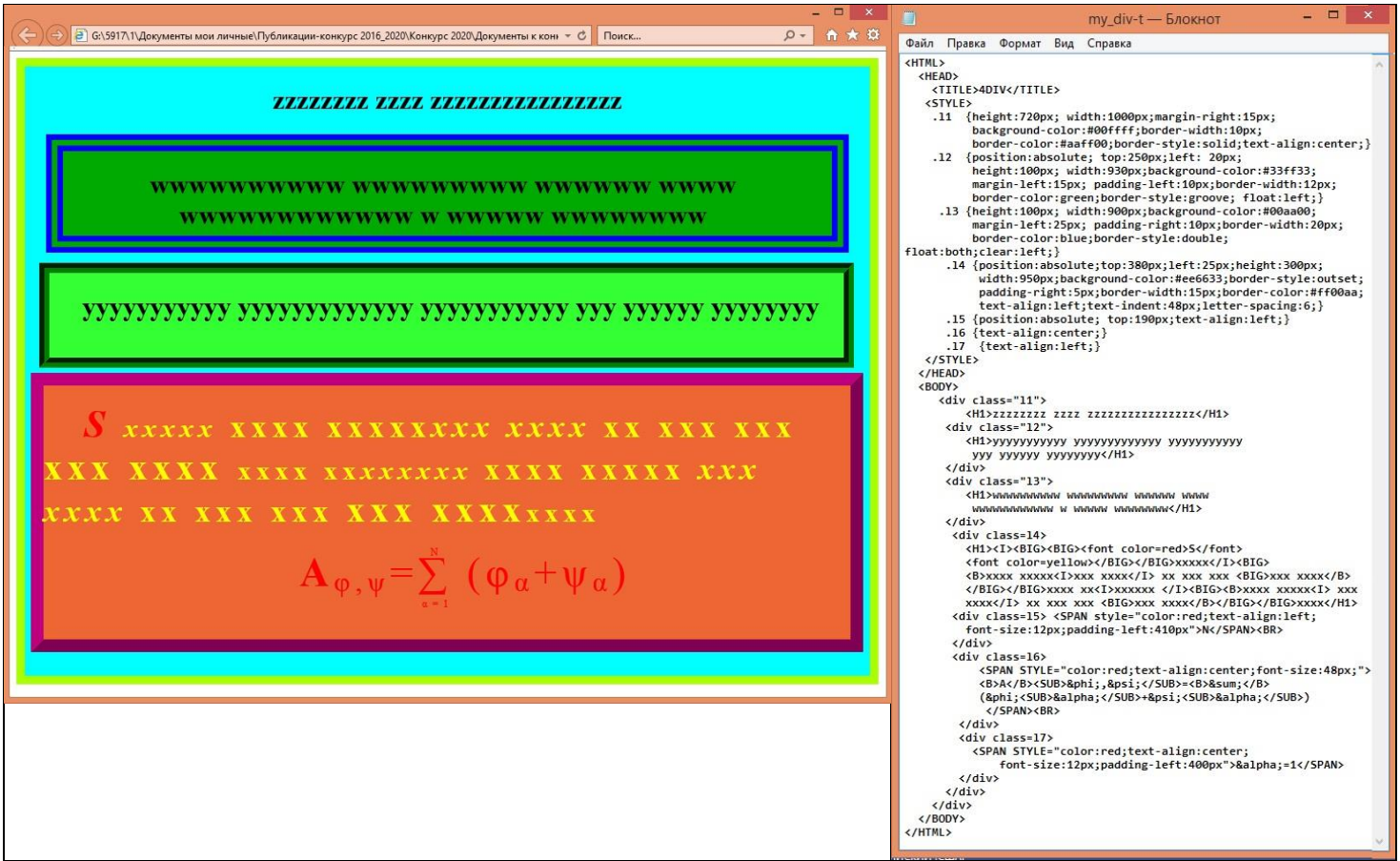


Рис. 33. Изображение и код web-страницы, соответствующие изображению на рис. 32, но с использованием описания стилей, размещённых внутри элемента ***STYLE***

Как видно из рисунков 32 и 33 одинаковые изображения можно получить, используя различные способы описания структуры web-страницы.

Код, представленный на рисунке 33, более предпочтителен, так как для большого набора страниц группировка тегов в виде элементов описания позволит повторно использовать их для различных фрагментов web-страниц.

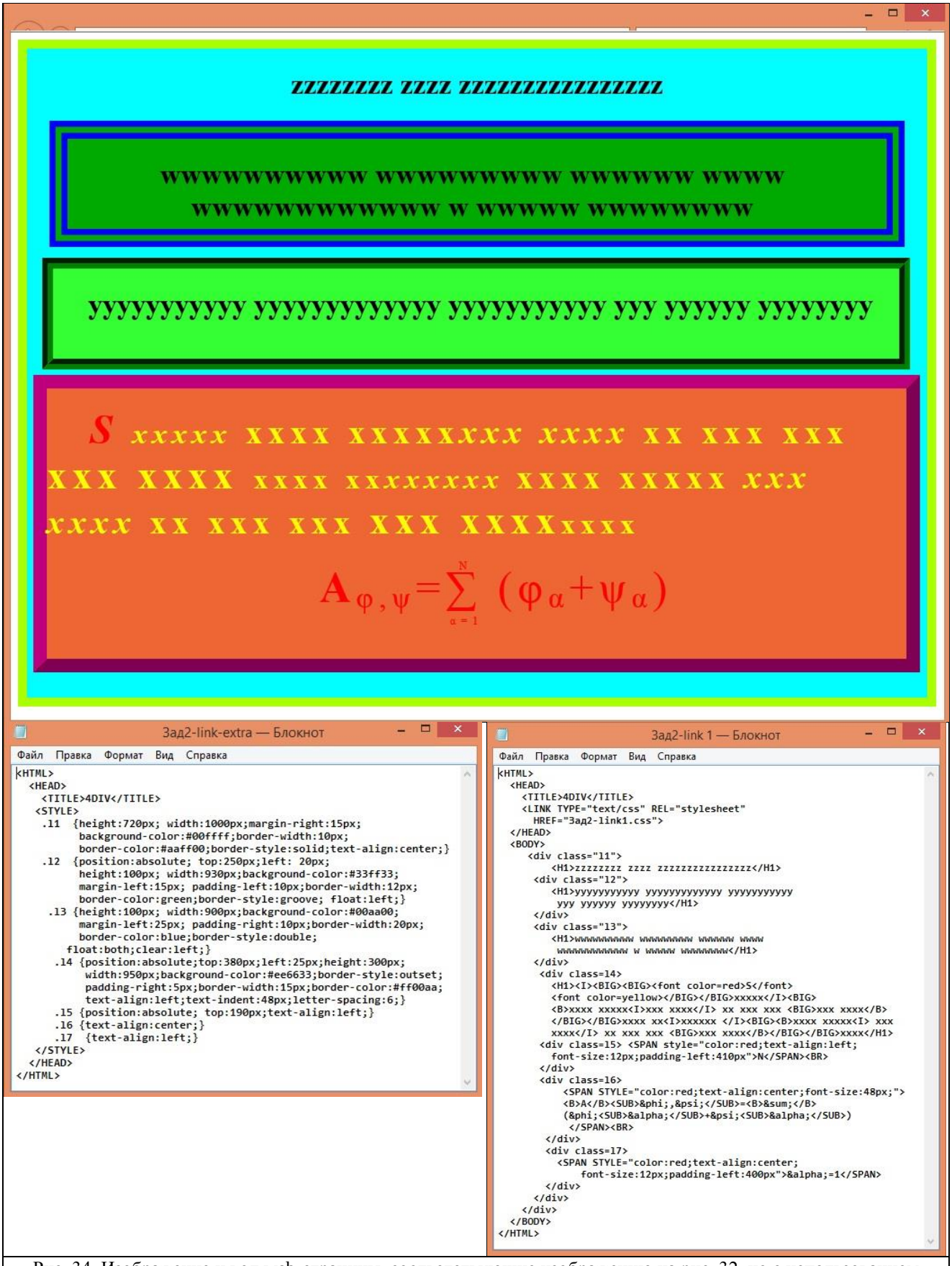


Рис. 34. Изображение и код web-страницы, соответствующие изображению на рис. 32, но с использованием описания стилей, размещённых в отдельном файле с расширением css (слева) и основной web-страницей (справа)

Попытайтесь творчески подойти к изучению селекторов каскадной таблицы стилей, научиться разным способам их организации. Добавьте изображения на web-страницу, сделайте их фоновым рисунком. Для того, чтобы шире изучить возможности CSS, проделайте задания из

методического электронного пособия ***css.pdf***. Задания 4-9 уже предполагают базовые знания языка JavaScript и могут демонстрировать псевдодвижение.

РАЗДЕЛ 2. Программирование на языке JavaScript

Тема 3. Основные приемы программирования на языке JavaScript (создание web-сценариев на базовом уровне)

В настоящем методическом пособии («Системное программирование. Первая часть») рассмотрим самые простые примеры сценариев с JavaScript.

Первые семь примеров, приведенных ниже и взятых из замечательной, методически удачно построенной книги талантливого преподавателя, доцента СПбГУ М. Дмитриевой «Самоучитель по JavaScript», наглядно представляют последовательные этапы абстракций в языке JavaScript от простого к сложному.

Для демонстрации особенностей языка сценариев JavaScript используются такие простейшие вычисления, как, например, вычисление площади прямоугольного треугольника. Целью же ниже предлагаемых примеров является методика обучения языку сценариев:

от использования только простых операторов (Задание 1.1), функции обычного вида (Задание 1.2), введения форм – важного и основного элемента объектно-ориентированного подхода (Задание 1.3 – Задание 1.6) с различным уровнем абстракций параметров функций, — до написания сценария без функций, но с формой и обработчиком событий в виде оператора присваивания

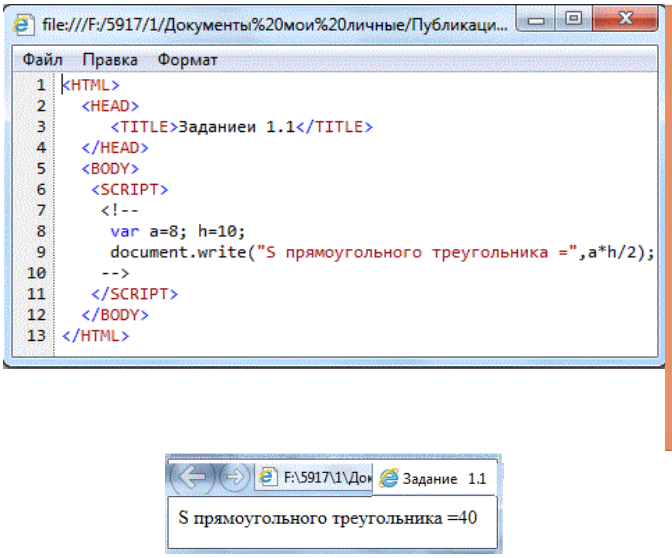


Рис. 35. Простейшая программа вычисления площади треугольника (код программы и изображение)

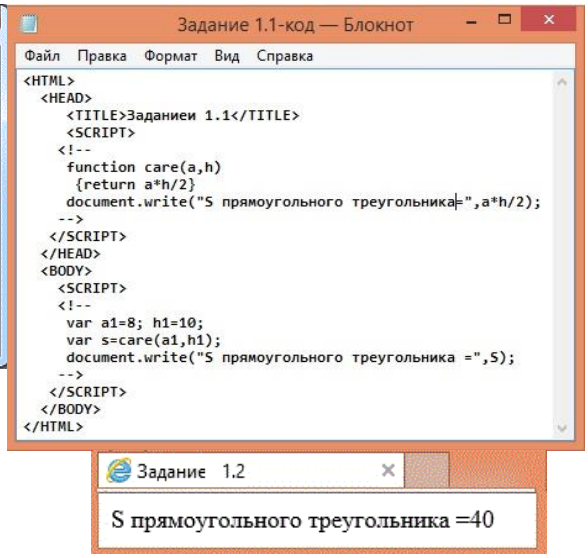


Рис. 36. Вычисление площади треугольника с использованием функции (код и изображение)

Сценарий с функцией

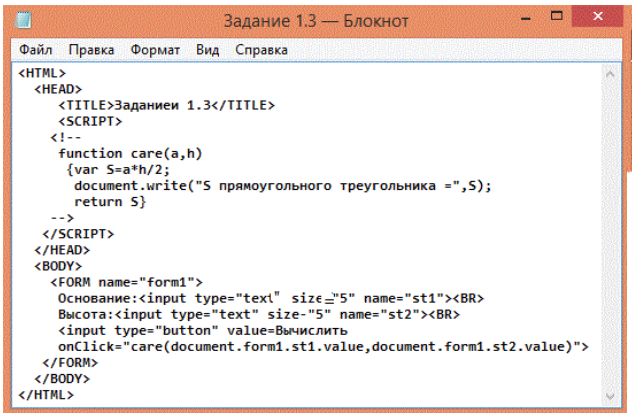


Рис. 37. Вычисление площади треугольника с помощью функции и формы (код программы)

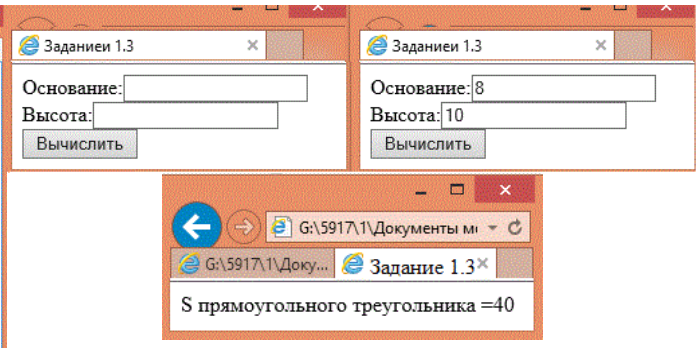


Рис. 38. Этапы изображения web-страницы: до ввода данных, после ввода данных и изображение результата

Обработка значений из формы

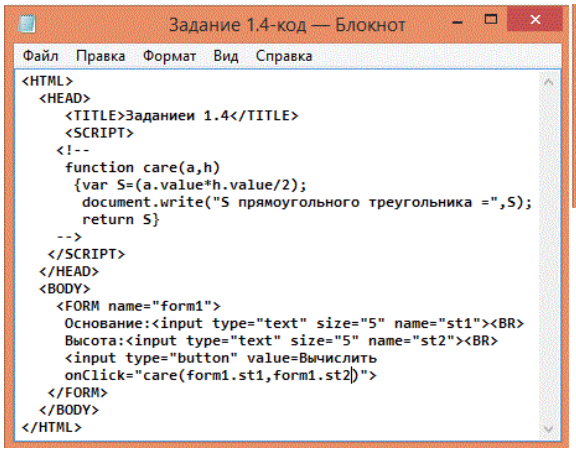


Рис. 39. Вычисление площади треугольника с функцией вызова по ссылке

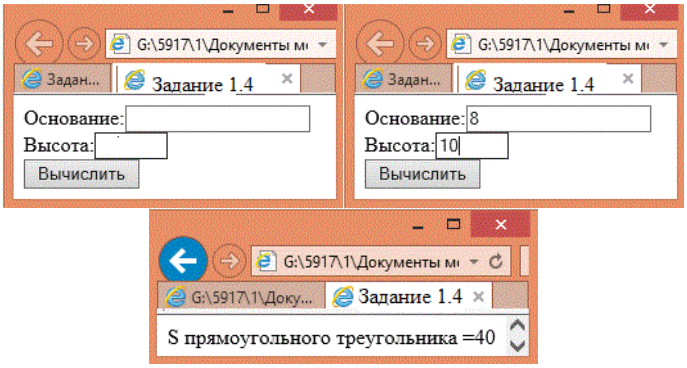


Рис. 40. Этапы изображения web-страницы: до ввода данных, после ввода данных и изображение результата

Передача параметров по ссылке

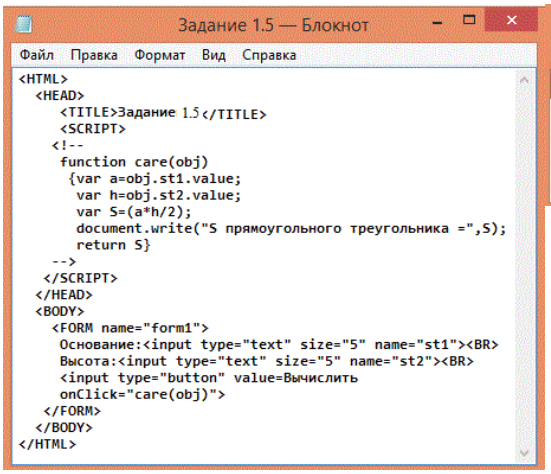


Рис. 41. Вычисление площади треугольника функцией с формой в качестве параметра

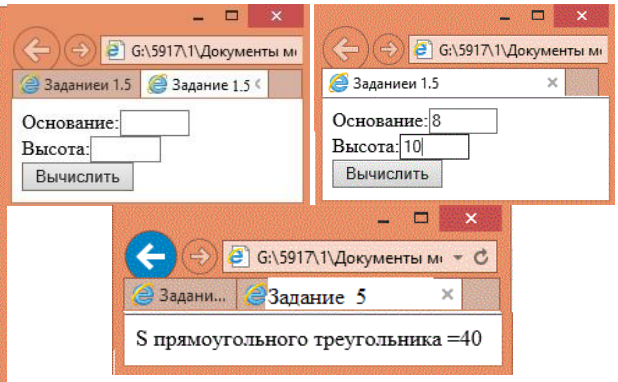


Рис. 40. Этапы изображения web-страницы: до ввода данных, после ввода данных и изображение результата

Использование имени формы в качестве параметра функции и представление ответа на отдельной странице

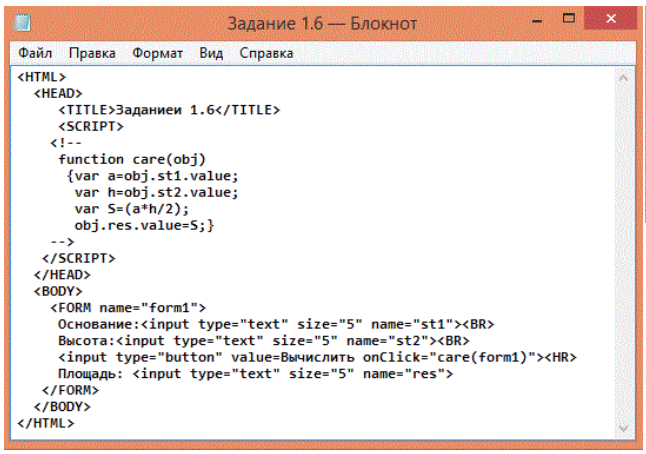


Рис. 43. Вычисление площади треугольника функцией с формой в качестве параметра

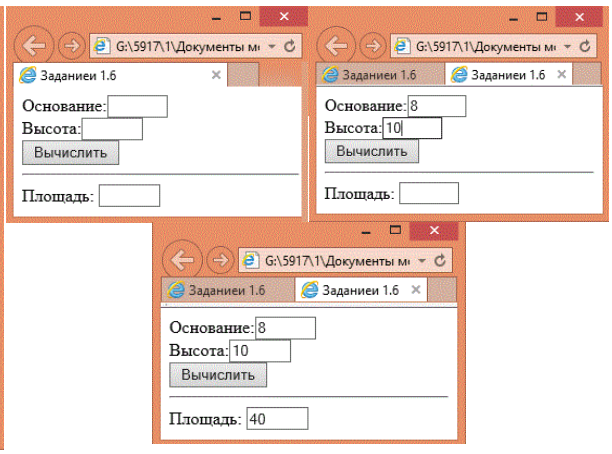


Рис. 44. Этапы изображения web-страницы: до ввода данных, после ввода данных и изображение результата

Использование имени формы в качестве параметра функции и представление ответа на той же странице

В вышеприведенных примерах (Задание 1.3 – Задание 1.6), а также в примере ниже (Задание 1.7) параметр обработки события ***onClick***, задающий действия, выполняемые при обработке события, был связан с элементом типа “кнопка” (***button***). Событие, вызывающее обработку элементов форм, выражается в щелчке мышью по кнопке с надписью “***Выполнить***”.

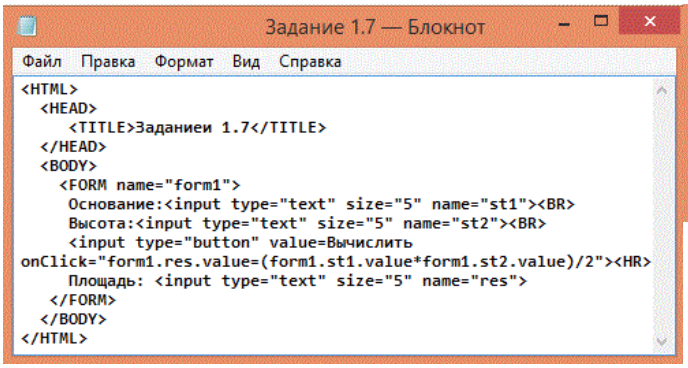


Рис. 45. Код с оператором присваивания

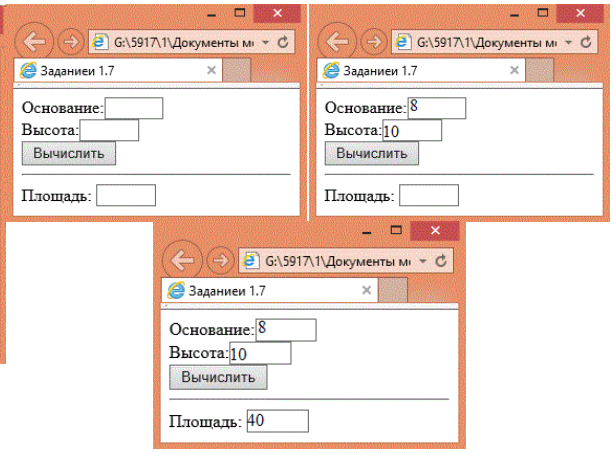


Рис. 46. Этапы изображения web-страницы: до ввода данных, после ввода данных и изображение результата

Использование оператора присваивания

В языке JavaScript существует целый список событий, которые по-разному обрабатывают элементы формы. Примеры обработки некоторых событий (различная реакция работы с данными при обработке элементов формы) рассмотрим на следующем примере.

Пример использования события ***Change***.

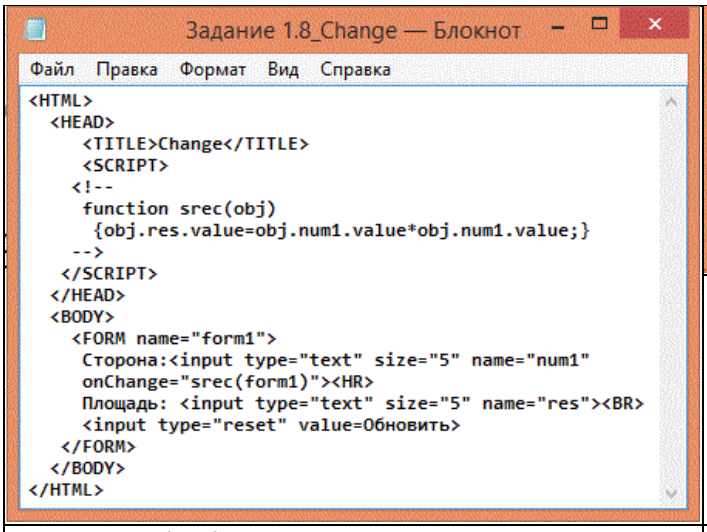


Рис. 47. Код вычисления площади квадрата с обработкой события ***Change***

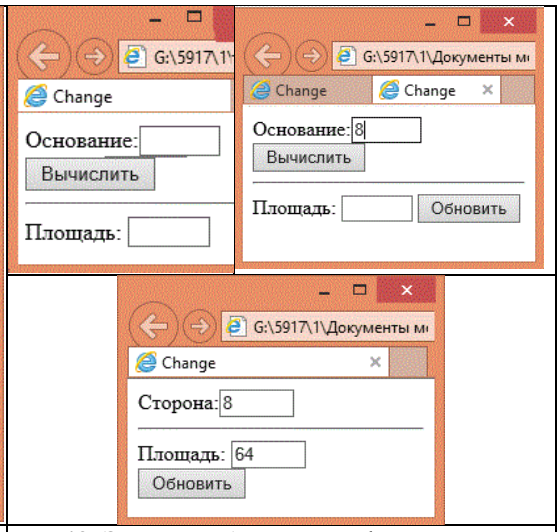


Рис. 48. Этапы изображения web-страницы: до ввода данных, после ввода данных и изображение результата

Обработка события ***Change***

В примере Задание 1.8 при обработке события ***Change*** после ввода данных(Сторона=8) для того, чтобы получить результат, нужно щелкнуть мышью вне поля данных (элемент потеряет фокус), тогда в поле Площадь появится результат 64. Кнопка “Обновить” служит для очистки всех полей (как на левом верхнем рисунке 48).

Если в коде программы (Рис. 47) заменить событие ***Change*** (параметр обработки события ***onChange***) на событие ***Focus***, ***Blur***, ***Select*** (соответственно, параметрами ***onFocus***, ***onBlur***, ***onSelect***), программа будет работать, однако, действия для получения результата будут различаться также, как и при замене на событие ***Click*** (***onClick***), с которым познакомились в Заданиях 1.3-1.7.

При обработке события ***Focus*** результат вычисляется в тот момент, когда пользователь переходит к элементу формы с помощью клавиши «***Tab***» или щелчком мыши по текстовому полю данных (Сторона). Очистив поле, можно задать другое значение стороны квадрата, и в тот момент, когда мышь наводится на текстовое поле данных (Сторона), то есть получает фокус, в поле результата появится соответствующее значение площади (результата вычисления).

Событие ***Blur*** (потеря фокуса) происходит в тот момент, когда элемент формы теряет фокус, то есть вычисления происходят в тот момент, когда поле данных теряет фокус, в этот момент в

поле результата помещается значение площади. Далее можно ввести новые данные в поле входных данных (Сторона), при переходе к любому другому элементу, в поле результата появится вычисленное значение (Площадь).

Событие ***Select*** вызывается выбором всего текстового поля (входных данных) или части этого текстового поля. При двойном щелчке мышью по этому текстовому полю исходное поле выделяется и наступает событие ***Select,*** обработка которого приводит к помещению результата вычисления площади квадрата в поле результата (Площадь).

Рассмотрим теперь следующее Задание, которое назовем Обмен (Обмен двух изображений), код и изображения которого представлены на рисунках 49-51.

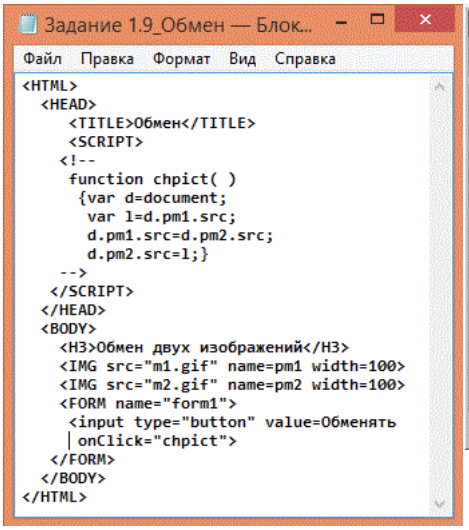


Рис. 49. Код программы обмена изображений

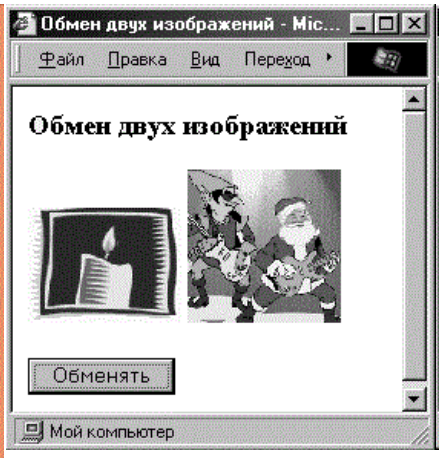


Рис. 50. Исходное изображение

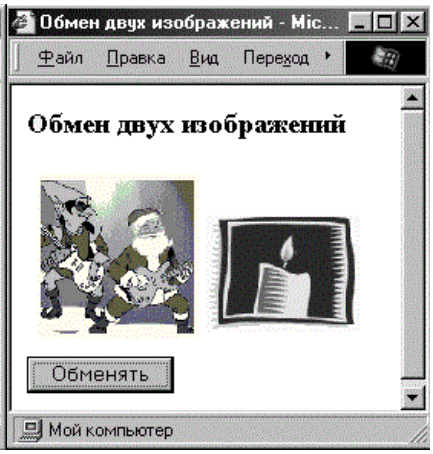


Рис. 51. Изображение после нажатия кнопки “Обменять”

Сценарий web-страницы обмена двух изображений

Обратите внимание, что для удобства

— каждому рисунку можно дать имя, например, ***name=pm1; name=pm2;***

— к адресу рисунка можно обратиться, например, следующим образом: ***d.pm1.src*** (объект ***document***->имя рисунка ***pm1***->адрес рисунка ***src***).

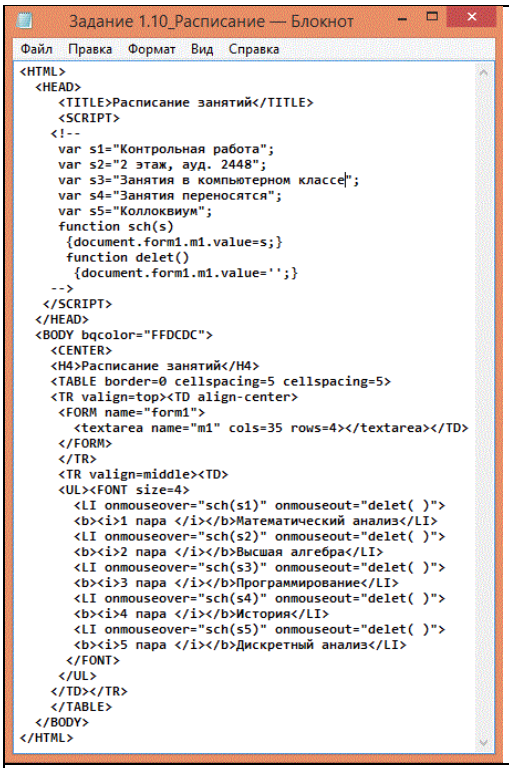


Рис. 52. Код сценария Расписание

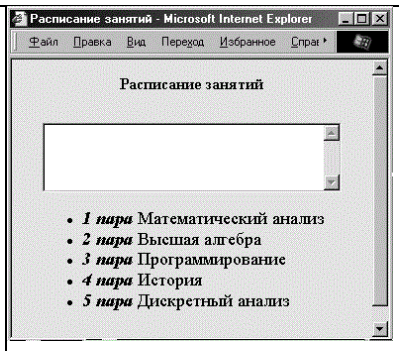


Рис. 53. Расписание. Начальное изображение

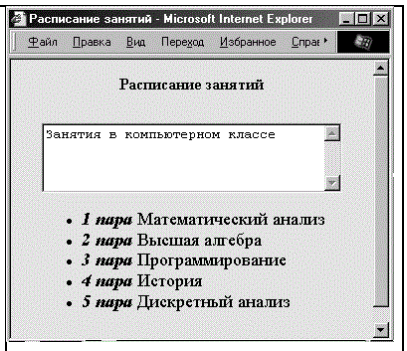


Рис. 54. Расписание. Изображение при наведении курсора на 3 пару

Выше приведенный пример — очень полезный пример, представляющий задание (Листинг 1.14) из учебника М. Дмитриевой, который называется **Расписание**. В этом задании используются события ***mouseover*** и ***mouseout***, действие первого состоит в том, что при наведении курсора мыши на объект (например, курсор над текстовым полем) выполняется одна запланированная реакция (появление текстовой строки в области), при смещении курсора с объекта (событие ***mouseout***) — выполняется другая реакция (удаление строки текста из области).

Для выполнения сценариев web-страниц, в которых производятся вычисления с использованием стандартных математических функций, необходимо использовать библиотеку ***Math: s=Math.sqrt(p*(p-a)*(p-b)*(p-c));***. Это задание не составляет проблем, поэтому подробно на нем останавливаться не будем.

Интересный пример представляет сценарий, реализующий циклическую смену изображений: по кнопке “Начать снова” осуществляется циклическая смена изображений через равные промежутки времени. По кнопке “Остановить” процесс останавливается.

Сценарий реализуется с использованием события ***load***, реакцией на это событие назначим вызов функции ***setTimeout***, которая и обеспечивает циклическую смену изображений. Напомним, что для простоты доступа, но не нарушая общности, изображения и сам сценарий должны находиться в одной папке.

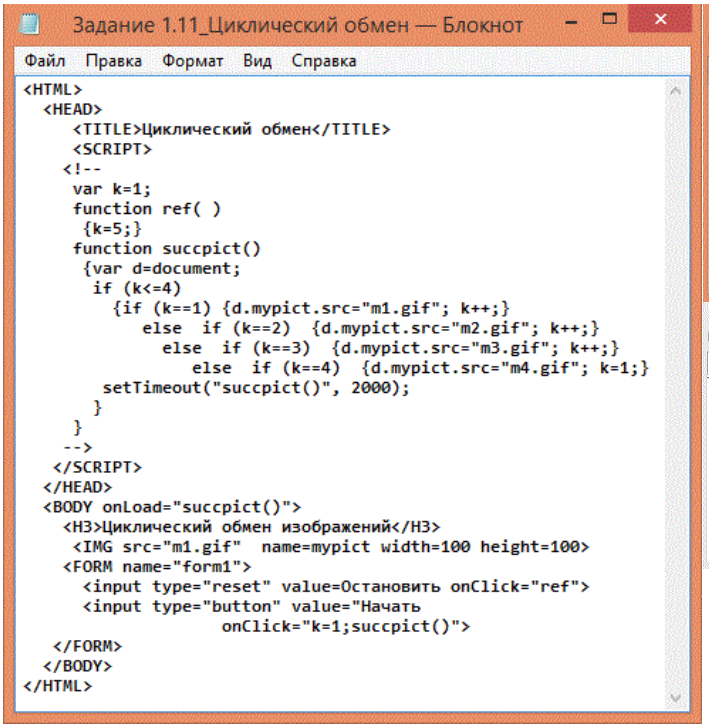


Рис. 55. Код сценария обмена изображений

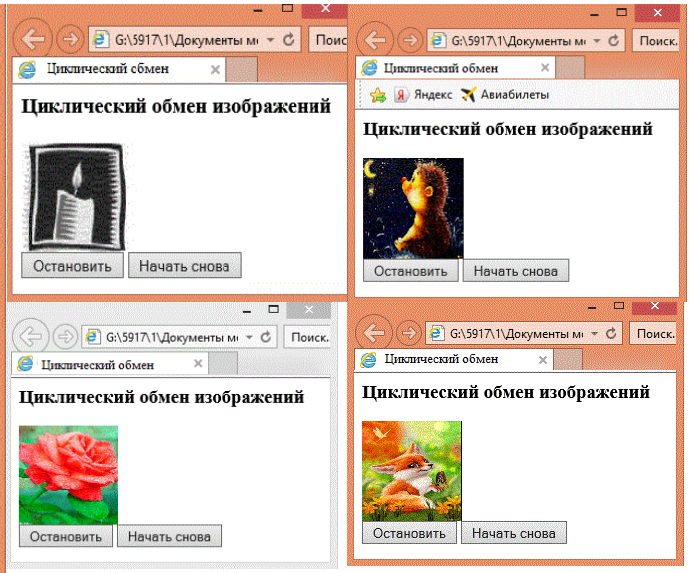


Рис. 56. Циклически сменяющиеся изображения (сверху вниз и слева направо)

Скорость смены изображений определяется вторым параметром функции ***setTimeout***:

setTimeout (“succpict( )”, 2000);

Число 2000 означает, что очередной вызов функции осуществляется через 2000 микросекунд, а результат вызова функции ***succpict( )*** — загрузка в окно браузера очередного рисунка.

Сохраняемые в папке рисунки могут быть любого размера, изображение этих рисунков на web-странице можно стандартизировать, указав в теге ***<IMG>*** атрибуты ***width*** и ***height:*** ***<IMG src=”m1.gif” width=100*** и ***height=100***>, при этом сами рисунки в папке свои заданные размеры не меняют.

Как Вы заметили, в настоящей методичке не рассматриваются все подряд задания по изучению языка программирования JavaScript, приведенные в списке рекомендованных для изучения дисциплины «Системное программирование» на базе учебного пособия [8]. Список рекомендованных заданий расположен после примеров по JavaScript перед Заключением (на стр. 27). Хотя каждое из рекомендованных заданий содержит полезные методические

особенности, объём методического пособия не позволяет рассмотреть их все, разберем логически связанные наиболее интересные.

Так, например, следующий пример (Эффект приближения изображения), показывает, как можно использовать функцию ***setTimeout*** совместно с изменением атрибутов HTML тегов.

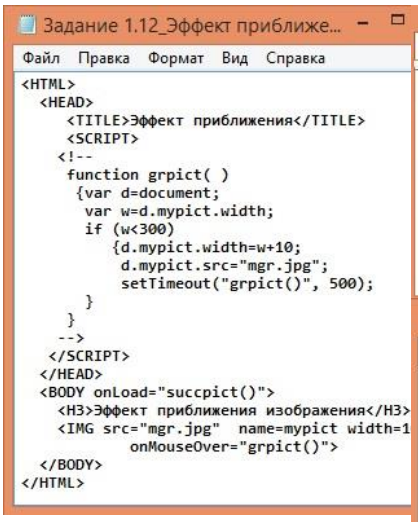


Рис. 57. Код сценария web-страницы приближения изображения

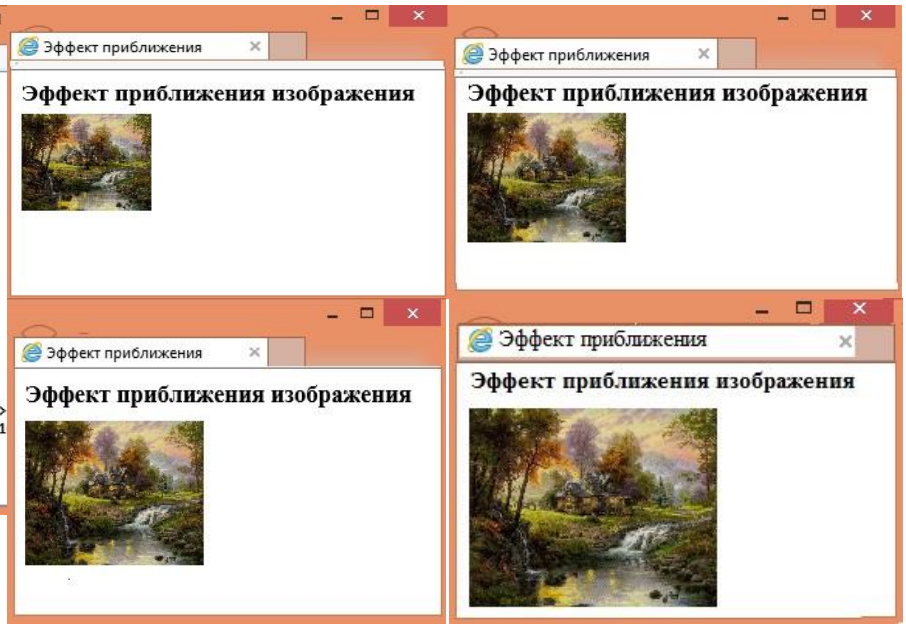


Рис. 58. Последовательные фрагменты изображений при выполнении сценария приближения изображения

Следующим примером проиллюстрируем сценарий, представляющий перестановку двух произвольно выбранных изображений из галереи четырех изображений.

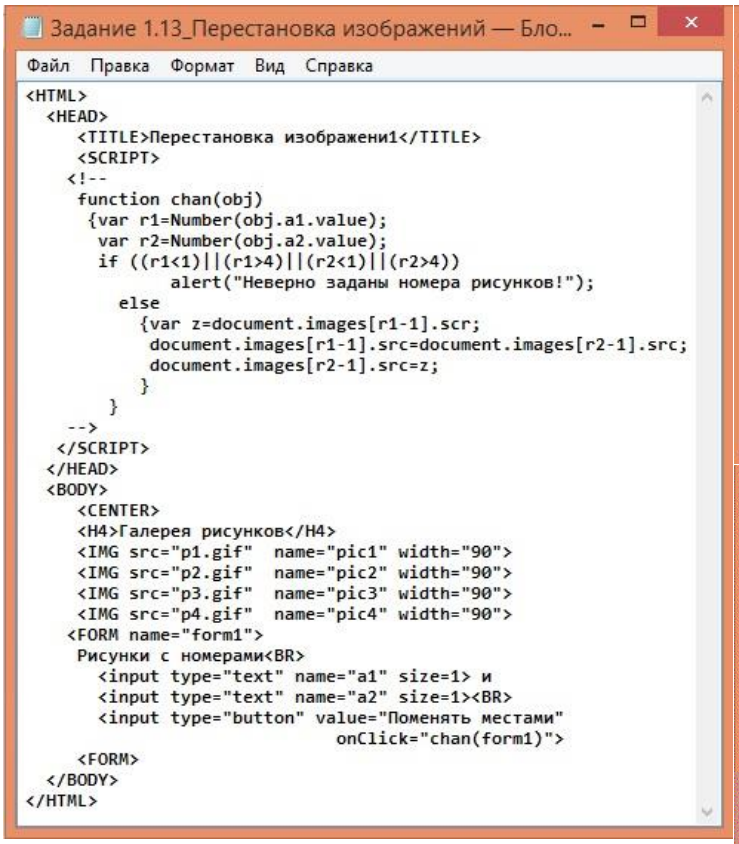


Рис. 59. Код сценария web-страницы обмена двух выбранных рисунков из галереи четырёх рисунков

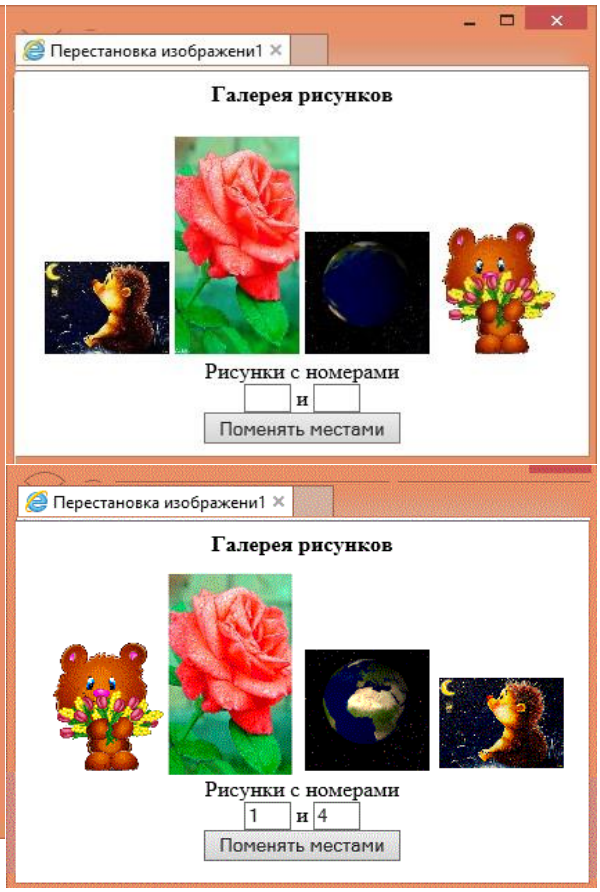


Рис. 60. Примеры обменов изображений попарно, соответственно выбранным номерам: верхнее изображение – исходное, нижнее – обмен 1 и 4

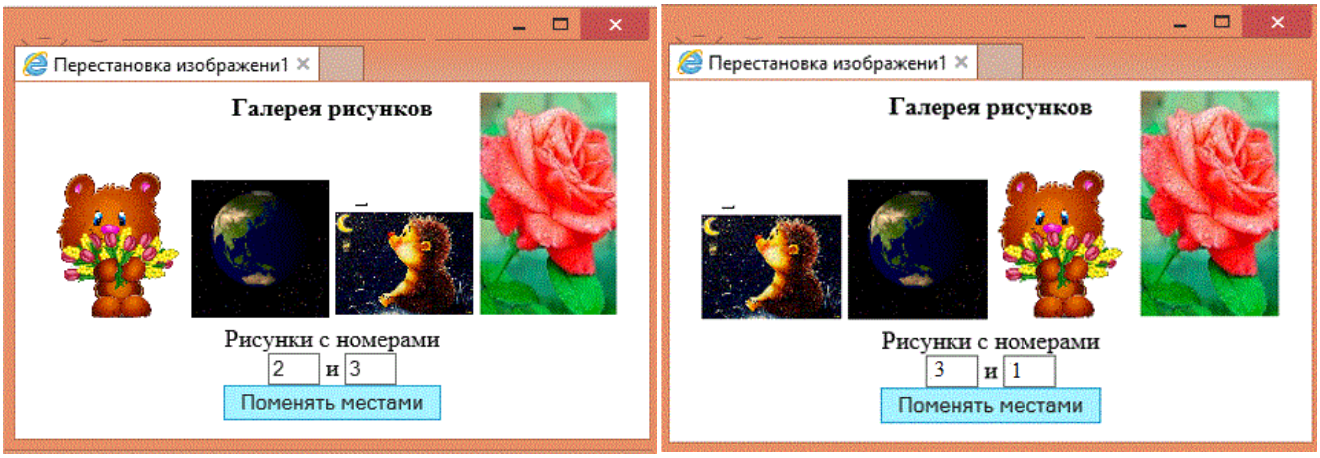


Рис. 61. Примеры обменов изображений попарно, соответственно выбранным номерам: левое изображение – обмен второго и третьего изображений, нижнее – обмен 1 и 3

В этом примере при обмене рисунков используются относительные адреса этих рисунков, присваемые переменным ***r1*** и ***r2***.

В качестве последнего примера (в настоящем методическом пособии) рассмотрим следующее задание. В задании требуется рассчитать нагрузку преподавателя по предмету в некоторой группе студентов и представить результаты в виде диаграммы.

Задание 1.14_Нагрузка преподавателя-1 — Блок...

Файл Правка Формат Вид Справка

```
<HTML>
  <HEAD>
     <TITLE>Нагрузка преподавателя</TITLE>
     <SCRIPT>
    <!--
     function graph(obj)
      {alert("A");
       var kl=0;alert("kl="+kl);
       var l=1*obj.lec.value;
       var p=1*obj.pract.value;
       var s=1*obj.stud.value;
       var tableHeight=150;
       var knTime=0;
       var eTime=0;
       var krTime=0;
       vat zTime=0;
       var d=document;
       if (s=="") alert("Введите количество студентов");
       if ((l!="")&&(l!=0))
         {knTime=Math.round(l*10/100);alert("knTime="+knTime);
          eTime=Math.round(s*30/60)};alert("eTime="+eTime);
       if ((p!="")&&(l!="0"))
          {knTime=Math.round(s*15/60);
           zTime=Math.round(s*20/60)};
        sTime = knTime+eTime+krTime+zTime;
        kl=tableHeight/sTime;
        d.images[0].height=Math.round(krTime*kl);
        d.images[0].alt=krTime+"часов";
        d.images[1].height=Math.round(zTime*kl);
        d.images[1].alt=zTime+"часов";
        d.images[2].height=Math.round(knTime*kl);
        d.images[2].alt=knTime+"часов";
        d.images[3].height=Math.round(eTime*kl);
        d.images[3].alt=eTime+"часов";
        d.images[4].height=Math.round(sTime*kl);
        d.images[4].alt=sTime+"часов";
    }
    function CheckValHous(hours)
        {if (hours<0)
         alert("Недопустимое значение кол-ва часов <0 !");
        }
     function CheckValNum(num)
        {if (num<=0)
         alert("Недопустимое значение <=0 !");
        }
    -->
   </SCRIPT>
  </HEAD>
```

Задание 1.14_Нагрузка преподавателя — Блокнот

Файл Правка Формат Вид Справка

```
  <BODY bgcolor=#crcccc>
     <CENTER>
     <H4>Анкета &quot;НАГРУЗКА ПРЕПОДАВАТЕЛЯ&quot;</H4>
     <SMALL>После заполнени нажать кнопку "Выполнить"</SMALL>
    <FORM name="data">
      <TABLE bgcolor=#cccccc width=300 cellPadding=2
                              cellSpacing=2 border=1>
        <TR><TD align=left><b>Лекции</b><SMALL>(часов)</SMALL></TD>
            <TD align=left><input type="text" name=lec size=5
                value="" onChange="CheckValHous(this.value)"></TD>
        </TR>
        <TR><TD align=left><b>Практика</b><SMALL>(часов)</SMALL></TD>
            <TD align=left><input type="text" name=pract size=5
                value="" onChange="CheckValHous(this.value)"></TD>
        </TR>
          <TR><TD align=left><b>Число студентов</b><SMALL>
                        (человек)</SMALL></TD>
            <TD align=left><input type="text" name=lec size=5
                value="" onChange="CheckValNum(this.value)"></TD>
        </TR>
         <TR align=center><TD colspan=2 bgcolor=#cccccc>
                    <input type="button" value="Вычислить"
              onClick="graph(data)"</TD>
          </TR>
         </TABLE>
       </FORM>
       <H4>Диаграмма нагрузки по вычисленным полям</H4>
     <TABLE width=212 height=190 cellPadding=0 cellSpacing=1 border=0>
       <TR valign=botton width=100 align=center>
         <TD bgcolor=#aaaaaa align=center>
           <IMG src=1x1.gif width-75 height=0 border=0></TD>
         <TD bgcolor=#aaaaaa align=center>
           <IMG src=1x1.gif width-75 height=0 border=0></TD>
         <TD bgcolor=#aaaaaa align=center>
           <IMG src=1x1.gif width-75 height=0 border=0></TD>
         <TD bgcolor=#aaaaaa align=center>
           <IMG src=1x1.gif width-75 height=0 border=0></TD>
         <TD bgcolor=#aaaaaa align=center>
           <IMG src=1x1.gif width-75 height=0 border=0></TD>
        </TR>
        <TR valign=top align=center wigth=100>
           <TD height=40><SMALL>Контрольная</SMALL></TD>
           <TD height=40><SMALL>Зачет</SMALL></TD>
           <TD height=40><SMALL>Консультация</SMALL></TD>
           <TD height=40><SMALL>Экзамен</SMALL></TD>
           <TD height=40><SMALL>Всего</SMALL></TD>
        </TR>
      </TABLE>
    </CENTER>
  </BODY>
</HTML>
```

Рис. 62. Код сценария, представляющего диаграмму нагрузки преподавателя по некоторой дисциплине для конкретного количества студентов

Входными данными для сценария являются количество часов лекций, часов практических занятий, а также количество студентов, определенных на данную дисциплину. Результатом

работы сценария окажется диаграмма, представляющая нагрузку преподавателя, вычисленную по стандартным алгоритмам.

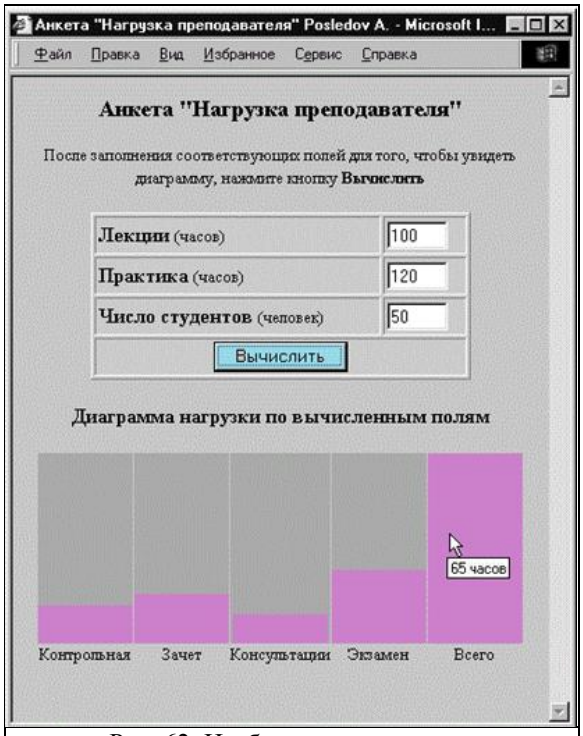


Рис. 62. Изображение сценария, представляющего нагрузку преподавателя

Ниже приводится рекомендуемый список заданий, которые помогут Вам получить необходимые первоначальные навыки и умения в области web-программирования. Список включает наиболее интересные и полезные с методической точки зрения задания. В настоящей же методичке разобраны лишь некоторые задания из приводимого списка.

Список заданий по JavaScript, которые необходимо выполнить для обучения программированию на языке программирования JavaScript.

Глава 1.

1.1, 1.2, 1.3, 1.4, 1.5, 1.5 с res, 1.6
1.7 (обратить внимание на ***a1=1*obj.num1.value;,*** что равносильно ***a1=Number(obj.num1.value);***
1.9 (*Focus*), 1.10 (*Blur*), 1.11 (*Select*),
1.12 (***d.pm1.value***),
1.14 (функция проявления надписи, событие ***onmouseover, onmouseout***),
1.15 (элемент формы ***textarea***, объект ***Math***)

Глава 2.

2.2 (***Math.min, Math.max, Number( );***), 2.4,
2.7 (функция ***setTimeout***),
2.8 ***(d.mypict.src="m"+k+".gif"***, событие внутри <***IMG***>),
2.9 (***d.mypict.width=w-10***)
2.13 (switch по числовой переменной),
2.15 (switch по строковой переменной),
2.16 (изображение), 2.17

Глава 3.

Информация об объектах: ***document.forms[i]***, ***document.images[i]*** и использование их атрибутов.
3.1а, 3.1б (доступ к свойствам с помощью ***document.forms[i].elements[i]***)
3.2 !,
3.5 (диаграмма), 3.6 (изменение параметров таблицы)

Глава 4. (radio)

4.1, 4.2 (использование строковой переменной для организации вывода),
4.3, 4.4 (выделенный элемент ***if ((d.form1.elements[i].checked))***
4.5 (выделенный элемент ***d.all("resch").value=k***),
4.7, 4.9

Глава 5. (checkbox)
5.1, 5.4 (параметр ***id***)
Глава 8.
8.4, 8.7,
Глава 10. (for in)
10.1
Глава 12.
12.1, 12.4, 12.6, 12.7

Глава 15. (eval)
Стр 327, 15.1, 15.2, 15.3, 15.4,
Глава 17.
17.1, 17.2, 17.3
Глава 19.
19.1, 19.2, 19.3, 19.5

Глава 6. (select)
6.1, 6.2, 6.5, 6.8, 6.9, 6.10
Глава 9.
9.3, 9.4,
Глава 11.
11.1, 11.2, 11.5, 11.7

Глава 14. (массивы)
14.2, 14.7, 14.9a, 14.9b (методы ***sort()*** и ***concat()***), 14.12
Глава 16. (рекурсия)
16.1, 16.2, 16.5, 16.6
Глава 18.
18.2, 18.3
Задания для зачета выбрать:
одно — из главы 19, вторую – из глав 13, 14, 15, третью – из глав 11, 12, 16, 17, 18

ЗАКЛЮЧЕНИЕ

Рассматриваемый материал представляет интерес для всех, кто связан с программированием приложений и служебных программ. Он определяет оптимальные пути получения обучающимися знаний и умений в соответствии с учебной программой дисциплины «Системное программирование». Отдельные части могут использоваться в таких курсах как «Информатика», «Практика на ЭВМ» и других дисциплинах, связанных с вычислительной техникой. Курс рассчитан на семестр (58 аудиторных часов и 46 часов самостоятельной работы), из которых большую часть составляют практические задания.

Практикум может быть полезен как преподавателям, так и обучающимся, осваивающим основы web-программирования с использованием языков HTML, CSS, JavaScript, а также создающим проекты и приложения на этих языках.

В методическом пособии приводится рекомендуемый список заданий, которые помогут студентам получить необходимые первоначальные навыки и умения в области web-программирования, а преподавателям составить план проведения практических заданий. Список включает наиболее интересные и полезные с методической точки зрения задания. В настоящем же методическом пособии подробно разобраны лишь некоторые задания по JavaScript из приводимого списка.

СПИСОК ЛИТЕРАТУРЫ

1. Г. Мейерс, Т. Баджетт, К. Сандлер. Искусство тестирования программ, М.: Диалектика. Вильямс, 2012.
2. Одинцов И. Профессиональное программирование. Системный подход СПб.: БХВ-Петербург, 2002.
3. Крис Джамса, Конрад Кинг, Энди Андерсон. Креативный Web-дизайн. HTML, XHTML, CSS, JavaScript, PHP, ASP, ActiveX. Текст, графика, звук и анимация. Изд.: ДиаСофт, 2005.
4. Прохоренок Н.А. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера Серия: Профессиональное программирование, Изд.: БХВ-Петербург, 2012.
5. В. В. Дунаев. HTML, скрипты и стили. СПб.: БХВ-Петербург, 2015.
6. В. А. Дронов. JavaScript в Web-дизайне. СПб.: БХВ-Петербург, 2005.
7. А. Ф. Холтыгин. Введение в HTML, CSS и JavaScript. СПб.: СПбГУ, 2006.
8. М. В. Дмитриева. Самоучитель JavaScript. СПб.: БХВ-Петербург, 2001.
9. В. А. Дронов. HTML 5, CSS 3 и Web 2.0. Разработка современных Web-сайтов. СПб.: БХВ-Петербург, 2011.
10. К. Шмитт. CSS. Рецепты программирования. М.: Русская редакция; СПб.: БХВ-Петербург, 2011.
11. П. Гастон. CSS3. Руководство разработчика. М.: Русская редакция; СПб.: БХВ-Петербург, 2012.
12. Э. Робсон, Э. Фримен. Изучаем HTML, XHTML и CSS. М.; СПб.; Нижний Новгород: Питер, 2015.
13. А. А. Дуванов. Web-конструирование. HTML. СПб.: БХВ-Петербург, 2005.
14. А. П. Сергеев. HTML и XML: Web-дизайн и программирование для среды Internet. М.: Диалектика, 2004.
15. М. Э. Хольцшлаг. Использование HTML и XHTML. М.: Вильямс, 2003.
16. Е. Л. Полонская. Язык HTML. М.: Вильямс, 2003.
17. М. Мак-Дональд. HTML5. Недостающее руководство. СПБ: БХВ-Петербург, 2015.
18. Э. Фримен, Э. Робсон. Изучаем программирование на JavaScript. М.; СПб.; Нижний Новгород: Питер, 2015.
19. С. А. Беляев. Разработка игр на языке JAVASCRIPT. СПб; М.; Краснодар: Лань, 2018.

ОГЛАВЛЕНИЕ