**20. Объясните механизм обработки событий с использованием DHTML.**

**HTML** (от англ. HyperText Markup Language, язык гипертекстовой разметки) - стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере. Веб-браузеры получают HTML документ от сервера по протоколам HTTP/HTTPS или открывают с локального диска, далее интерпретируют код в интерфейс, который будет отображаться на экране монитора.

**DHTML** (динамический HTML) – это набор средств, которые позволяют создавать более интерактивные Web-страницы без увеличения загрузки сервера. Другими словами, определенные действия посетителя ведут к изменениям внешнего вида и содержания страницы без обращения к серверу.

DHTML построен на объектной модели документа (Document Object Model, DOM), которая расширяет традиционный статический HTML-документ. DOM обеспечивает динамический доступ к содержимому документа, его структуре и стилям. В DOM каждый элемент Web-страницы является объектом, который можно изменять. DOM не определяет новых тегов и атрибутов, а просто обеспечивает возможность программного управления всеми тегами, атрибутами и каскадными листами стилей (CSS).

На WEB-страницах можно обеспечить реакцию на определенные действия посетителя или изменения состояния документа или окна, которые вызывают определенные события.

Посетитель генерирует события при передвижении мыши, нажатии кнопок мыши и клавиатуры. Изменения состояния документа генерируют события при загрузке документа, изображений или объектов, при появлении ошибки на странице или переходе фокуса от одного элемента к другому.

Модель событий DHTML связана с определенной иерархией HTML-контейнеров и основана на всплывании событий и действии по умолчанию.

Всплывание событий заключается в том, что на событие может быть получена реакция не только от элемента-источника события, но также и от всех его родительских элементов вплоть до тела документа и самого документа. Событие может быть обработано на любом уровне.

Действие по умолчанию обеспечивается встроенной в броузер обработкой событий. Например, действием по умолчанию на нажатие ссылки <a href="…"> является переход по указанному адресу и загрузка страницы. Многие события позволяют заменить встроенные действия по умолчанию на индивидуальную обработку.

Установление связи между определенным событием и сценарием называется связыванием событий. События можно связать с помощью специальных атрибутов любого элемента или с помощью тега script.

Связывание событий с атрибутами удобно, но требует расширения языка HTML каждый раз при изобретении нового события. А так как HTML развивается медленно, данный подход используется только для небольшого набора встроенных событий. Как атрибуты любого элемента в DHTML представлены события мыши и клавиатуры.

Применение script является более общим механизмом связывания события со сценарием. Все события представлены как свойства в DOM.

События мыши

**OnMouseOver**

Перемещение указателя мыши на элемент.

**OnMouseOut**

Перемещение указателя мыши за пределы элемента.

**OnMouseDown**

Нажатие любой кнопки мыши. Внутри обработчика event.button указывает, какая кнопка нажата: 1 = левая, 2 = правая, 4 = средняя.

**OnMouseUp**

Отпускание любой кнопки мыши. Внутри обработчика event.button указывает, какая кнопка отпущена: 1 = левая, 2 = правая, 4 = средняя.

**OnMouseMove**

Перемещение указателя мыши. Внутри обработчика event.x и event.y – текущие координаты "горячей" точки курсора на экране.

**OnClick**

Щелчок левой кнопкой мыши на элементе или нажатие <Enter> при каком-то элементе в фокусе.

**OnDblClick**

Двойной щелчок левой кнопкой мыши на элементе.

**OnDragStart**

Запуск операции перетаскивания. Цель – запретить операцию перетаскивания путем возвращения значения false.

**OnSelectStart**

Запуск операции выделения элемента. Цель – запретить выделение области документа. Важно учесть, что отменяется лишь инициализация выделения, т.е. если выделение начато за пределами данной области, а на ней лишь продолжается, то помешать выделению нельзя. Событие **OnSelectStart** всплывающее, поэтому можно перехватить его и вернуть значение false. Это лишит посетителя возможности выделять текст на странице.

**OnSelect**

Выделение элемента. Следует за **OnSelectStart** и возникает много раз по мере того, как посетитель расширяет или сужает выделение. Событие **OnSelect** не всплывает. Оно возникает лишь в том разделе документа, где происходит выделение.

События клавиатуры

**OnKeyPress**

Нажатие и отпускание клавиши. Событие возникает много раз, если клавиша удерживается.

**OnKeyDown**

Нажатие клавиши. Событие возникает один раз, даже если клавиша продолжает удерживаться.

**OnKeyUp**

Отпускание клавиши.

Событие прокручивания

**OnScroll**

Использование полосы прокрутки или прокручивание элемента неявно посредством другого действия. Не может отменить прокручивания, так как возникает после завершения прокручивания. Не всплывает.

**OnFocus**

Возникает, когда элемент активизируется после щелчка по нему мышью или с помощью клавиатуры. Фокус могут получить только элементы пользовательского ввода и тело документа, а не элементы содержания документа.

**OnBlur**

Возникает, когда элемент теряет фокус. Используется для контроля корректности ввода.

**OnChange**

Возникает при любых изменениях в форме. Для текстового поля это ввод или удаление. Для списка – выбор нового элемента. Так же, как и **OnBlur** используется для контроля корректности ввода.

**OnSubmit**

Возникает перед отправкой содержимого формы на сервер. Основное назначение – проверка корректности введенных данных. Отмена передачи данных происходит при возвращении значения **false**.

**OnReset**

Возникает после щелчка на кнопке **Reset**. Назначение – вывод предупреждения о сбросе.

События документа

**OnLoad**

Возникает после считывания, но до отображения страницы. Используется для отображения заставки, проверки права доступа и других однократных действий при инициализации страницы.

**OnUnload**

Возникает при выгрузке страницы. Используется для освобождения каких-либо ресурсов и вывода дополнительных сообщений.

**OnAbort**

Возникает при срыве загрузки графического объекта.

**OnError**

Возникает лишь при неудачной загрузке графического элемента или всей WEB-страницы. Все остальные ошибки это событие не отлавливает.

**OnHelp**

Запрос файла справки с использованием клавиши <F1>. Не возникает при выборе пункта Help из меню Help. **OnHelp** возникает на элементе с фокусом и затем всплывает вверх по иерархии контейнеров. Действие по умолчанию для этого события заключается в отображении встроенного файла справки, но данное событие может быть отменено для отображения индивидуального файла справки.

Для получения информации о событии служит объект **event**.

Свойства объекта event

**srcElement**

Элемент, который первым сгенерировал событие. Имя тега этого элемента можно определить с помощью свойства **tagName**.

**cancelBubble**

Используется для прекращения всплывания события вверх по иерархии контейнеров. По умолчанию равно **false** и событие всплывает. Установка значения **true**останавливает всплывание текущего события.

**returnValue**

Используется для отмены действия по умолчанию. Для этого нужно установить значение **false**. Ключевое слово **return** в JavaScript обновляет значение **event.returnValue**при возвращении управления броузеру.

**altKey**

Состояние клавиши **<Alt>**(true/false).

**ctrlKey**

Состояние клавиши **<Ctrl>**(true/false).

**shiftKey**

Состояние клавиши **<Shift>**(true/false).

**keyCode**

ASCII-код нажатой клавиши.

Динамическое содержание

Содержание HTML-документа можно менять после загрузки страницы.

Свойства динамического содержания

**innerText**

Содержащийся внутри элемента текст без тегов. При установке нового значения все HTML-теги выводятся как текст.

**innerHTML**

Содержащийся внутри элемента текст вместе с внутренними тегами. При установке нового значения все HTML-теги интерпретируются броузером.

**outerText**

Содержащийся внутри элемента текст без тегов. При установке нового значения меняется весь элемент, включая внешние теги. Нестандартное свойство! Не стоит использовать, т.к. не поддерживается браузером Firefox.

**outerHTML**

Весь элемент, включая внешние теги. При установке нового значения все HTML-теги интерпретируются броузером.

Данные свойства динамического содержания могут обращаться к элементу или его содержанию только как к целому. Если, например, требуется изменить лишь один символ в параграфе, то придется заново переписать весь параграф. Однако, можно напрямую манипулировать любой частью HTML-документа. Такую возможность предоставляет объект TextRange.

Методы Adjacent

Кроме перечисленных четырех свойств динамического содержания имеются еще два метода для вставки содержания:

**insertAdjacentText**

Вставка текста.

**insertAdjacentHTML**

Вставка HTML.

Оба метода Adjacent имеют по два аргумента: место вставки и содержание. Место вставки определяется ключами:

**beforeBegin** – До открывающего внешнего тега элемента

**afterBegin** – После открывающего тега до содержимого элемента

**beforeEnd** – До закрывающего тега после содержимого элемента

**afterEnd** – После закрывающего тега элемента.

Методы Adjacent удобно использовать для вставки новых параграфов или элементов списков.

Объект TextRange

Используя объект **TextRange** можно редактировать любой текст, а также менять текст, выделенный посетителем на экране.

Объект TextRange создается путем вызова метода **createTextRange** элемента, который является, так называемым, владельцем редактирования текста. Это может быть **body**, **input**, **button** и **textarea**, причем понятно, что **body** - владелец редактирования текста для всех воспроизводимых элементов, а **input**, **button** и **textarea** - владельцы редактирования текста для своего содержания.

Свойства доступа к тексту

**text**

Текст документа без HTML-разметки. Это свойство доступно для чтения/записи. По представлению содержания сходно с outerText.

**htmlText**

Текст документа вместе с HTML-разметкой. По представлению содержания это свойство сходно с outerHTML, но доступно только для чтения.

Метод вставки HTML

**pasteHTML**

При вызове этого метода для определенного элемента вставляемый фрагмент будет находиться внутри области действия данного элемента и должен быть HTML-кодом внутри данной области действия. По представлению содержания сходно с outerHTML.

Методы позиционирования объекта TextRange

**collapse([prm1])**

Сворачивает объект TextRange к началу (prm1=true) или к концу элемента (prm1=false). Значение по умолчанию prm1=true.

**expand(prm1)** Расширяет объект TextRange для полного охвата определенного параметром фрагмента текста.  
Параметр prm1 может принимать значения:

**character** – символ,

**word** – слово,

**sentence** – предложение.

**moveToElementText(prm1)**

Размещает объект TextRange для охвата текста элемента, указанного параметром prm1.

**move(prm1, prm2)**

Перемещает начало объекта TextRange на расстояние, указанное параметрами, и сжимает объект до точки. Параметр prm1 для этого и двух следующих методов может принимать те же значения, что и для метода expand. Параметр prm2 может принимать как положительные, так и отрицательные значения; указывает расстояние перемещения.

**moveStart(prm1, prm2)**

Перемещает начало объекта TextRange на расстояние, указанное параметрами.

**moveEnd(prm1, prm2)**

Перемещает конец объекта TextRange на расстояние, указанное параметрами.

Методы **move**, **moveStart** и **moveEnd** возвращают число, равное расстоянию перемещения. Если будет задано перемещение на 100 слов в документе, содержащим 50 слов, то объект TextRange будет помещен в последнее слово, а метод возвратит расстояние перемещения. Для проверки успешности проведения перемещения возвращаемое значение сравнивается с заданным расстоянием перемещения.

**moveToPoint(prm1, prm2)**

Определяет элемент, который выведен в точке, координаты которой заданы параметрами prm1, prm2. Работает не совсем корректно. В чем можно убедиться на примере.

**findText(prm1, prm2, prm3)**

Поиск в документе подстроки, указанной в параметре prm1. Параметр prm2 определяет число символов для поиска (положительное - прямое направление, отрицательное - обратное). Параметр prm3 может принимать значения:

**2** – полное совпадение,

**4** – учет регистра,

**6** – полное совпадение с учетом регистра.

Методы управления объектом TextRange

**duplicate**

Создает копию объекта TextRange, из которого он был вызван (var tr1=document.body.createTextRange();  var tr2=tr1.duplicate(); ).

**inRange**

Определяет нахождение указанной области внутри объекта TextRange, из которого был вызван данный метод (alert(tr2.inRange(tr1)); // true).

**isEqual**

Сравнивает два объекта TextRange для определения, охватывают ли они один и тот же текст (alert(tr1.isEqual(tr2)); // true ). Метод необходим, так как прямое сравнение для разных объектов, представляющих одну и ту же текстовую область работать не будет ( alert(tr1==tr2); // false).

**compareEndPoints(prm1, prm2)**

Сравнивает два объекта TextRange для определения, совпадают ли их начальные и конечные позиции. Метод вызывается из первого объекта. Второй объект определяется параметром prm2. Параметр prm1 определяет, что метод сравнивает. Возможные значения prm1:

**StartToStart** – начало первого с началом второго,

**StartToEnd** – начало первого с концом второго,

**EndToStart** – конец первого с началом второго,

**EndToEnd** – конец первого с концом второго.

**setEndPoints(prm1, prm2)**

Устанавливает начальную и конечную позиции одного объекта TextRange в начальную и конечную позиции другого объекта TextRange. Метод вызывается из первого объекта. Второй объект определяется параметром prm2. Параметр prm1 определяет, что метод устанавливает. Возможные значения prm1:

**StartToStart** – начало первого в начало второго,

**StartToEnd** – начало первого в конец второго,

**EndToStart** – конец первого в начало второг,о

**EndToEnd** – конец первого в конец второго.

Например, чтобы определить, будет ли продолжен объект trSecond после окончания объекта trOne, надо написать такую функцию:  
function continue(trOne, trSecond)  
{ return trOne.compareEndPoints("EndToStart", trSecond); }.

**49. Название фирмы - заголовок первого уровня, выравнивается по центру, буквы голубого цвета.**

Остальной текст выравнивается по правому краю, шрифт синего цвета, начертание - курсив, размер шрифта на 3 единицы меньше текущего.

Таблица расположена по центру, с заголовком, расположенным над таблицей, ширины рамки 2 пикселя, ячейки, в которых расположен заголовок, серого цвета.

Список вложенный. Типы маркеров выбрать самостоятельно.

Для задания свойств цвета и фона использовать внешнюю таблицу стилей.

**<!doctype html>**

**<html>**

**<head>**

**<meta charset="utf-8">**

**<title>1</title>**

**<style type="text/css">**

**table {**

**background: grey; /\* Цвет фона таблицы \*/**

**color: white; /\* Цвет текста \*/**

**}**

**td {**

**background: grey; /\* Цвет фона ячеек \*/**

**color: white;**

**}**

**</style>**

**</head>**

**<h2 align="center" style="color:skyblue">Контора обувщик</h2>**

**<body>**

**<font color="blue">**

**<h4 align="right">Делаем обувь на века</h4> <br></font>**

**<table border="2" align="center">**

**<caption>Таблица размеров обуви</caption>**

**<tr>**

**<th>Россия</th>**

**<th>Великобритания</th>**

**<th>Европа</th>**

**<th>Длина ступни, см</th>**

**</tr>**

**<tr><td>34,5</td><td>3,5</td><td>36</td><td>23</td></tr>**

**<tr><td>35,5</td><td>4</td><td>36⅔</td><td>23–23,5</td></tr>**

**<tr><td>36</td><td>4,5</td><td>37⅓</td><td>23,5</td></tr>**

**<tr><td>36,5</td><td>5</td><td>38</td><td>24</td></tr>**

**<tr><td>37</td><td>5,5</td><td>38⅔</td><td>24,5</td></tr>**

**<tr><td>38</td><td>6</td><td>39⅓</td><td>25</td></tr>**

**<tr><td>38,5</td><td>6,5</td><td>40</td><td>25,5</td></tr>**

**<tr><td>39</td><td>7</td><td>40⅔</td><td>25,5–26</td></tr>**

**<tr><td>40</td><td>7,5</td><td>41⅓</td><td>26</td></tr>**

**<tr><td>40,5</td><td>8</td><td>42</td><td>26,5</td></tr>**

**<tr><td>41</td><td>8,5</td><td>42⅔</td><td>27</td></tr>**

**<tr><td>42</td><td>9</td><td>43⅓</td><td>27,5</td></tr>**

**<tr><td>43</td><td>9,5</td><td>44</td><td>28</td></tr>**

**<tr><td>43,5</td><td>10</td><td>44⅔</td><td>28–28,5</td></tr>**

**<tr><td>44</td><td>10,5</td><td>45⅓</td><td>28,5–29</td></tr>**

**<tr><td>44,5</td><td>11</td><td>46</td><td>29</td></tr>**

**<tr><td>45</td><td>11,5</td><td>46⅔</td><td>29,5</td></tr>**

**<tr><td>46</td><td>12</td><td>47⅓</td><td>30</td></tr>**

**<tr><td>46,5</td><td>12,5</td><td>48</td><td>30,5</td></tr>**

**<tr><td>47</td><td>13</td><td>48⅔</td><td>31</td></tr>**

**<tr><td>48</td><td>13,5</td><td>49⅓</td><td>31,5</td></tr>**

**</table>**

**<ol>**

**<li>Кожаные ботинки</li>**

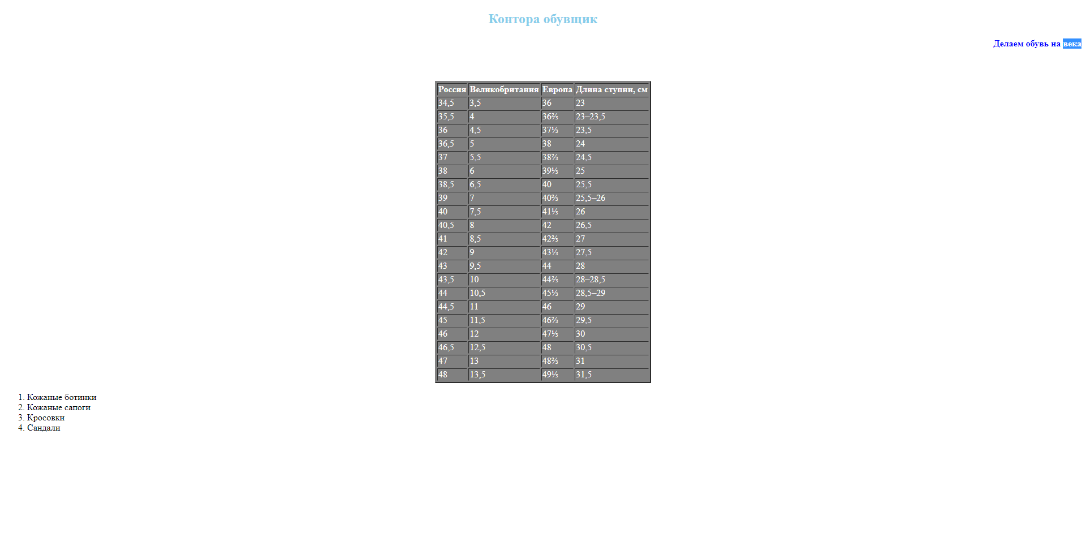
**<li>Кожаные сапоги </li>**

**<li>Кросовки</li>**

**<li>Сандали</li></ol>**

**</body>**

**</html>**

****

**80 Требуется создать три скрипта на языке JavaScript и подключить их одной HTML-странице.**

Первый скрипт позволяет разместить в строке заголовка браузера Ваши фамилию, имя и отчество. Установить время вывода заголовка и длительность паузы до начала очередного вывода заголовка в соответствии с Вашим вариантом по таблице 1.

Второй скрипт содержит список, при выборе элемента которого цвет фона страницы меняется на выбранный. Список должен содержать не менее трех элементов. Цвета выбрать самостоятельно.

Третий скрипт предназначен для проверки знания таблицы истинности логического элемента. Тип логической функции, форму ввода информации и вид запускающего проверку события выбрать из таблицы 2 в соответствии с вариантом. Для обозначения событий приняты следующие сокращения: ЩЛК - щелчок левой кнопкой мыши (onClick); ДЩК - двойной щелчок левой кнопкой мыши (onDbClick); НКК - наведение курсора мыши на кнопку (onMouseOver)

Таблица 1 – Варианты заданий для первого скрипта

| **Номер**  **варианта** | **Время вывода** | **Пауза** |
| --- | --- | --- |
| 1 | 2 | 3 |
| 61 | 2 секунды | 3 секунды |
| 62 | 3 секунды | 2 секунды |
| 63 | 4 секунды | 4 секунды |
| 64 | 5 секунд | 2 секунды |
| 65 | 6 секунд | 3 секунды |
| 66 | 3 секунды | 4 секунды |
| 67 | 4 секунды | 3 секунды |
| 68 | 5 секунд | 2 секунды |
| 69 | 2 секунды | 4 секунды |
| 70 | 3 секунды | 5 секунд |
| 71 | 4 секунды | 3 секунды |
| 72 | 5 секунд | 4 секунды |
| 73 | 2 секунды | 3 секунды |
| 74 | 3 секунды | 2 секунды |
| 75 | 4 секунды | 4 секунды |
| 76 | 5 секунд | 2 секунды |
| 77 | 2 секунды | 3 секунды |
| 78 | 3 секунды | 4 секунды |
| 79 | 4 секунды | 3 секунды |
| 80 | 6 секунд | 2 секунды |

Таблица 2 - Варианты заданий для третьего скрипта

| **Номер**  **варианта** | **Логическая функция** | **Форма** | **Событие** |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 61 | A ∧ B | Кнопка | ЩЛК |
| 62 | A ∧ (B ∨ C) | Переключатель | ДЩК |
| 63 | A ∨ ( B ∧ C) | Включатель | НКК |
| 64 | ⎤А ∧ B | Список | ЩЛК |

Продолжение таблицы 2

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 65 | ⎤А ∨ B | Кнопка | ДЩК |
| 66 | ⎤ (A ∧ B) | Переключатель | НКК |
| 67 | A ∧ B ∨ C | Включатель | ЩЛК |
| 68 | A ∧ B ∧ ⎤C | Список | ДЩК |
| 69 | ⎤А ∧ B | Кнопка | НКК |
| 70 | A ∨ ( B ∧ C) | Кнопка | ЩЛК |
| 71 | A ∧ B ∧ ⎤C | Переключатель | ЩЛК |
| 72 | A ∧ B | Включатель | ДЩК |
| 73 | ⎤(A ∧ B) | Список | НКК |
| 74 | A ∧ B ∧ ⎤C | Кнопка | ЩЛК |
| 75 | A ∧ B ∨ C | Список | ДЩК |
| 76 | A ∨ ( B ∧ C) | Кнопка | НКК |
| 77 | ⎤А ∧ B | Включатель | ДЩК |
| 78 | A ∨ ( B ∧ C) | Переключатель | ЩЛК |
| 79 | A ∧ (B ∨ C) | Включатель | НКК |
| 80 | ⎤А ∨ B | Переключатель | ЩЛК |

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>zadanie2</title>

</head>

<body>

<script> function dothing(){

let name =document.querySelector('input[name=Name]').value;

let sname = document.querySelector('input[name=Suname]').value;

let otchestvo = document.querySelector('input[name=Father]').value;

setInterval(() =>{document.title = "pauza"}, 2000);;

setInterval(() => {document.title = sname +" " + name +" " + otchestvo}, 6000);

}

function changeColor(colorParam) {

let color = colorParam.value.toLowerCase();

var optionElement = document.getElementById('rgb');

document.body.style.backgroundColor = color;

};

function logicthing() {

let x = parseInt(document.querySelector('input[name=log]:checked').value);

let z = parseInt(document.querySelector('input[name=seclog]:checked').value);

let yXor = x ^ z;

document.title = yXor.toString();

};

</script>

<table width = "300" border="0" bordercolor = "#000000" cellpadding = "2" cellspacing = "0">

<tr>

<td align = "left">Фамилия:</td>

<td><input name="Suname" style="color:white; background-color: #99AAFF; font-style: oblique" required placeholder=""></td>

</tr>

<tr>

<td align = "left">Имя:</td>

<td><input name="Name" style="color:white; background-color: #99AAFF; font-style: oblique" required placeholder=""></td>

</tr>

<tr >

<td align = "left">Отчество:</td>

<td><input name="Father" style="color:white; background-color: #99AAFF; font-style: oblique" required placeholder=""></td>

</tr>

</table><br/>

<input id="clickMe" type="button" value="clickme" onclick="show()" />

<select onchange="changeColor(this);" class="color" id="rgb">

<option id="red" value="Red">Red</option>

<option id="green" value="Green">Green</option>

<option id="blue" value="Blue">Blue</option>

<option id="white" value="White">White</option>

<option id="pink" value="Pink">Pink</option>

</select>

<p><input name="log" type="radio" value="1"> 1</p>

<p><input name="log" type="radio" value="0" > 0</p>

<p><input name="seclog" type="radio" value="1" > 1</p>

<p><input name="seclog" type="radio" value="0" > 0</p>

<p><input type="submit" value="Выбрать" onclick="logicthing()" ></p>

</body>

</html>

**94 Спроектировать структуру базы данных о списке рассылки и подписчиках: тема и содержание письма, дата отправки, имена и адреса подписчиков, их учетные записи и пароли.**

**Запросы:**

**а) вывести информацию о рассылках, принадлежащих определенной теме;**

**б) вывести информацию о рассылках, отправленных в определенный период времени.**

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

**<html>**

**<head>**

**<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">**

**<title>Вывод данных</title>**

**</head>**

**<body>**

**<fieldset>**

**<form method="post" action="select\_user.php">**

**<label for="first\_name">Тема Рассылки</label>**

**<br/>**

**<input type="text" name="first\_name" size="30"><br/>**

**<input id="submit" type="submit" value="Найти и вывести"><br/>**

**</form>**

**<form method="post" action="select\_date.php">**

**<label for="last\_name">Дата Рассылки</label>**

**<br/>**

**<input type="text" name="last\_name" size="30"><br/>**

**<input id="submit" type="submit" value="Найти и вывести"><br/>**

**</form>**

**</fieldset>**

**<fieldset>**

**</fieldset>**

**</body>**

**</html>**

**<html>**

**<head>**

**<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">**

**<title>results</title>**

**</head>**

**<body>**

**<?php**

**$connt = mysqli\_connect("localhost", "root", "","users");**

**$first\_name = trim($\_REQUEST['last\_name']);**

**$sql\_select = "SELECT \* FROM users WHERE date='$first\_name'";**

**$result = mysqli\_query($connt,$sql\_select);**

**while ($row = mysqli\_fetch\_array($result)) {**

**$field1name = $row["keysp"];**

**$field2name = $row["date"];**

**$field3name = $row["topic"];**

**$field4name = $row["soderzhanie"];**

**$field5name = $row["name"];**

**$field6name = $row["adress"];**

**$field7name = $row["username"];**

**$field8name = $row["password"];**

**echo '<table border="1" cellspacing="5" cellpadding="5" width="100%"><tr>**

**<td>'.$field1name.'</td>**

**<td>'.$field2name.'</td>**

**<td>'.$field3name.'</td>**

**<td>'.$field4name.'</td>**

**<td>'.$field5name.'</td>**

**<td>'.$field6name.'</td>**

**<td>'.$field7name.'</td>**

**<td>'.$field8name.'</td>**

**</tr></table>';**

**}**

**?>**

**<a href="zadanie3.html">return to search</a><br/><br/>**

**</body>**

**</html>**

**<html>**

**<head>**

**<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">**

**<title>results</title>**

**</head>**

**<body>**

**<?php**

**$connt = mysqli\_connect("localhost", "root", "","users");**

**$first\_name = trim($\_REQUEST['first\_name']);**

**$sql\_select = "SELECT \* FROM users WHERE topic='$first\_name'";**

**$result = mysqli\_query($connt,$sql\_select);**

**while ($row = mysqli\_fetch\_array($result)) {**

**$field1name = $row["keysp"];**

**$field2name = $row["date"];**

**$field3name = $row["topic"];**

**$field4name = $row["soderzhanie"];**

**$field5name = $row["name"];**

**$field6name = $row["adress"];**

**$field7name = $row["username"];**

**$field8name = $row["password"];**

**echo '<table border="1" cellspacing="5" cellpadding="5" width="100%"><tr>**

**<td>'.$field1name.'</td>**

**<td>'.$field2name.'</td>**

**<td>'.$field3name.'</td>**

**<td>'.$field4name.'</td>**

**<td>'.$field5name.'</td>**

**<td>'.$field6name.'</td>**

**<td>'.$field7name.'</td>**

**<td>'.$field8name.'</td>**

**</tr></table>';**

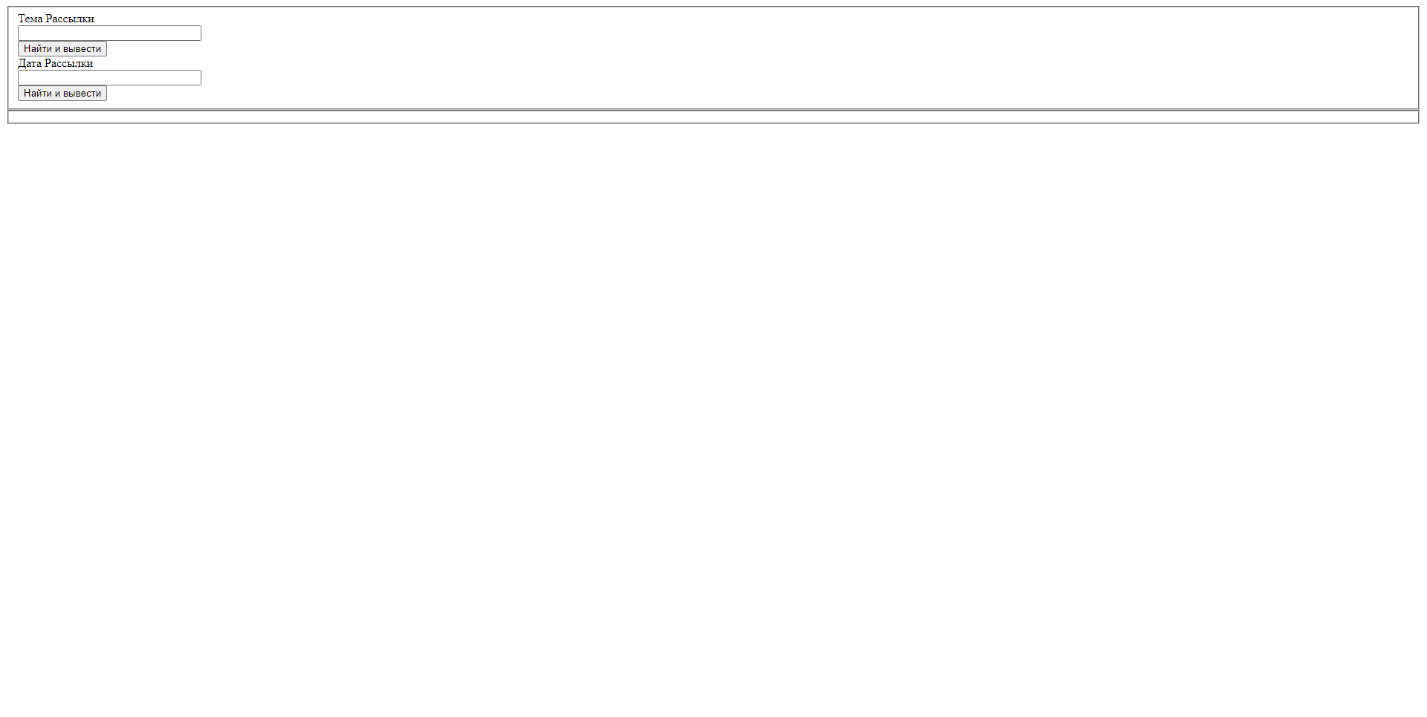
**}**

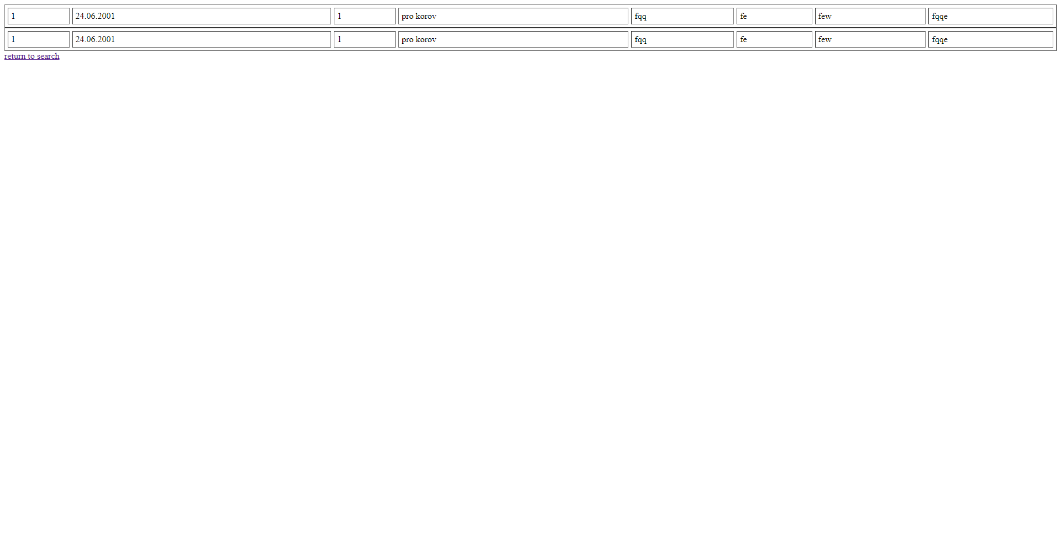
**?>**

**<a href="zadanie3.html">return to search</a><br/><br/>**

**</body>**

**</html>**

****

****