Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра программного обеспечения
информационных технологий

С. С. Куликов

БАНКОВСКИЕ ИНТЕРНЕТ-ТЕХНОЛОГИИ

Методическое пособие
к лабораторным работам
для студентов специальности
«Программное обеспечение информационных технологий»
всех форм обучения

Минск БГУИР 2012

УДК 004.9:336.71(076.5)
ББК 32.973.26-018.2+65.262.1я73
К90

Р е ц е н з е н т ы:
доцент кафедры менеджмента технологий
Института бизнеса и менеджмента технологий Белорусского государственного университета, кандидат физико-математических наук
А. Ф. Смалюк;

доцент кафедры сетей и устройств телекоммуникаций учреждения образования
«Белорусский государственный университет информатики
и радиоэлектроники»,
кандидат технических наук О. Г. Смолякова

Куликов, С. С.

К90 Банковские интернет-технологии: метод. пособие к лаб. работам для студ. спец. «Программное обеспечение информационных технологий» всех форм обуч. / С. С. Куликов. – Минск : БГУИР, 2012. – 32 с. : ил.

ISBN 978-985-488-853-8.

В пособии представлены восемь лабораторных работ, в которых рассмотрены основные подходы к построению интернет-ориентированных приложений. Приведены рекомендации по выполнению работ, примеры.

УДК 004.9:336.71(076.5)
ББК 32.973.26-018.2+65.262.1я73

ISBN 978-985-488-853-8

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данное методическое пособие представляет собой руководство по установке и настройке необходимого программного обеспечения и выполнению лабораторных работ.

Лабораторные работы представлены в двух видах: стандартном, предназначенном для общего использования, и расширенном, предназначенном для студентов, обладающих углубленными знаниями в предметной области дисциплины «Банковские интернет-технологии». Выбор стандартного или расширенного варианта лабораторных работ осуществляется студентом самостоятельно. Стандартный вариант лабораторных работ подразумевает выполнение одного из вариантов заданий, который назначается преподавателем.

Выполнение лабораторных работ подразумевает использование свободно распространяемого программного обеспечения, а именно:

– веб-сервер – Apache;
– СУБД – MySQL;
– среда исполнения и язык программирования – PHP;
– средство проектирования БД – phpMyAdmin;
– среда разработки программ на PHP – Notepad++.

Основная терминология, используемая в данном пособии

Веб-клиент – приложение, выполняемое на локальной рабочей станции пользователя и устанавливающее соединения с веб-сервером по мере необходимости обмена данными. Некоторые операции могут выполняться на стороне клиента в том случае, если они не требуют информации с сервера и ориентированы в основном на работу с пользователем. Наиболее распространёнными веб-клиентами являются браузеры.

Веб-сервер – программное обеспечение, получающее HTTP-запросы от клиентов (см. веб-клиент) и генерирующее ответы, которые, как правило, представляют собой HTML-документы и связанные с ними данные (графические изображения, файлы CSS, XML и т. п.) Наиболее известными веб-серверами являются Apache, Lighttpd, Nginx, Miscrosoft IIS.

Среда исполнения – программное окружение, изолирующее приложение, написанное на некотором высокоуровневом языке программирования, от операционной системы и аппаратного обеспечения. Основная задача среды исполнения – обеспечение переносимости приложений между программными и аппаратными платформами. Наиболее известными средами исполнения являются: PHP, Java Runtime Environment (JRE), Microsoft .NET Framework.

В разделе «Установка и настройка необходимого программного обеспечения» приведено подробное описание процесса подготовки рабочего места для выполнения лабораторных работ. Основной упор сделан на придание рабочему месту свойств реальных хостинговых платформ, чем обусловлено использование отдельных программных средств вместо готовых «пакетов веб-ПО».

Установка и настройка необходимого программного обеспечения

Для выполнения лабораторных работ необходимо загрузить из сети Интернет следующее программное обеспечение.

Веб-сервер Apache (версии 2.2.x или новее) по адресу:
http://projects.apache.org/projects/http_server.html
СУБД MySQL (версии 5.1.x, или 5.5.x или новее) по адресу:
http://dev.mysql.com/downloads/
Среду исполнения PHP (версии 5.3.x или новее) по адресу:
http://php.net/
Среду проектирования БД phpMyAdmin (версии 3.3.x или новее) по адресу:
http://www.phpmyadmin.net
Средство разработки программ на PHP Notepad++ (версии 5.8.x или новее) по адресу:
http://notepad-plus-plus.org/download

Последовательность установки программного обеспечения такова.

1. Перед установкой веб-сервера Apache необходимо выполнить из командной строки команду "telnet 127.0.0.1 80" и убедиться, что соединение не может быть установлено. Это означает, что 80-й порт, по которому будет работать Apache, свободен. После этого установите веб-сервер Apache со всеми настройками по умолчанию.

2. Установите PHP, указав в опциях инсталлятора, что PHP должен работать как модуль веб-сервера Apache. Обязательно укажите, что следует использовать следующие расширения: mysql, mysqli, gd, mbstring.

3. Установите MySQL со всеми настройками по умолчанию.

4. Установите Notepad++ со всеми настройками по умолчанию.

5. Теперь необходимо правильно настроить Apache и PHP.

В файле настроек Apache httpd.conf измените значение параметра DocumentRoot на c:/www, предварительно создав такую папку. Также замените на c:/www все пути, совпадающие со старым значением DocumentRoot. Измените значение параметра DirectoryIndex: перед index.html добавьте index.php.

В файле настроек PHP php.ini установите следующие значения параметров: short_open_tag = On, output_buffering = Off, max_execution_time = 30, max_input_time = 60, memory_limit = 128M, error_reporting = E_ALL, display_errors = On, post_max_size = 64M, upload_max_filesize = 64M, session.save_path="C:\WINDOWS\Temp" (удостоверьтесь, что такая папка существует!), date.timezone = 'Europe/Minsk'

6. Распакуйте содержимое архива с дистрибутивом phpMyAdmin в папку c:/www/ phpmyadmin/.

7. Перезагрузите компьютер. Всё готово. Также вы можете посмотреть видеоинструкцию по установке в материалах курса.

Лабораторная работа №1. Основы HTML и CSS

Цель работы: изучение основ языков гипертекстовой разметки HTML и управления визуальным оформлением HTML CSS.

Порядок выполнения работы

1. Изучить темы 2.1–2.4 лекционного материала.
2. Выполнить задание по лабораторной работе.
3. Представить для проверки результат выполнения работы в виде файлов HTML, CSS, JS (в зависимости от варианта задания).

Стандартное задание

Вариант 1: разработать веб-страницу (HTML, CSS), содержащую форму, представленную на рисунке 1. Использовать табличную вёрстку.

Вариант 2: разработать веб-страницу (HTML, CSS), содержащую форму, представленную на рисунке 2. Использовать блочную вёрстку.

Рисунок 1 – Форма для варианта 1

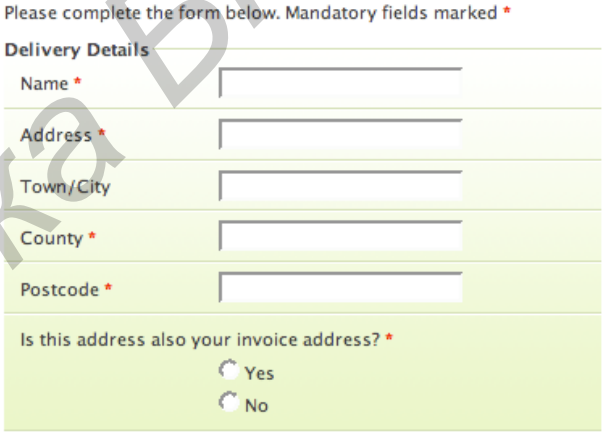

Рисунок 2 – Форма для варианта 2

Вариант 3: разработать титульную веб-страницу (HTML, CSS) сайта со структурой, показанной на рисунке 3. Наполнить страницу произвольным содержимым. Использовать табличную вёрстку.

Вариант 4: разработать титульную веб-страницу (HTML, CSS) сайта со структурой, показанной на рисунке 4. Наполнить страницу произвольным содержимым. Использовать блочную вёрстку.

Вариант 5: разработать форму оформления заказа в интернет-магазине. Набор полей, вёрстка и применяемые технологии – произвольные.

Вариант 6: разработать форму регистрации в сервисе бесплатной почты. Набор полей, вёрстка и применяемые технологии – произвольные.

Вариант 7: разработать веб-страницу со списком новостей, в котором для каждой новости отображается заголовок, дата публикации, автор, краткая аннотация. Применяемые технологии – произвольные. Использовать блочную вёрстку.

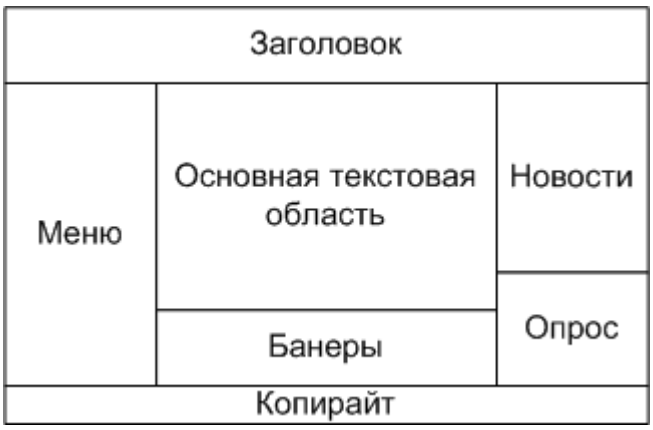

Рисунок 3 – Страница для варианта 3

Рисунок 4 – Страница для варианта 4

Вариант 8: разработать веб-страницу со списком сотрудников некоторой гипотетической организации, в котором для каждой позиции списка отображается ФИО сотрудника, фотография, дата рождения, дата принятия на работу, должность, некоторая дополнительная текстовая информация. Применяемые технологии – произвольные. Использовать блочную вёрстку.

Вариант 9: разработать веб-страницу со списком товаров гипотетического интернет-магазина, в котором для каждой позиции списка отображается название товара, фотография, стоимость, основные характеристики, некоторая дополнительная текстовая информация. Применяемые технологии – произвольные. Использовать табличную вёрстку.

Вариант 10: разработать веб-страницу с фотогалереей. Для каждого изображения вывести информацию об авторе, линейных размерах изображения и размерах файла в килобайтах. Применяемые технологии – произвольные. Использовать блочную вёрстку.

Расширенное задание

Разработать набор HTML-страниц сайта одной из следующих тематик: блог, интернет-магазин, новостной сайт, сайт-визитка, сайт государственной организации, сайт учебного заведения, сайт музыкальной группы, сайт интернет-провайдера, сайт оператора сотовой связи, сайт банка. Применяемые технологии и тип вёрстки – произвольные. Каждый набор должен включать не менее пяти страниц разного типа (например: титульная, новости, поиск, карта сайта, каталог товаров и тому подобное).

Рекомендации по выполнению заданий

В процессе выполнения данной лабораторной работы особое внимание следует уделить корректности HTML- и CSS-кода и вопросам совместимости с различными браузерами. Изменение контента (добавление и удаление текста, изображений и тому подобное) не должно приводить к нарушению структуры HTML-страниц. По возможности следует уделить внимание вопросам дизайна, цветового оформления и использования графических элементов, а также вопросам удобства использования (юзабилити).

Лабораторная работа №2. Основы PHP

Цель работы: изучение основ языка программирования PHP, работы с переменными, операторами, условными конструкциями, циклами.

Порядок выполнения работы

1. Изучить темы 3.1–3.7 лекционного материала.
2. Выполнить задание по лабораторной работе.
3. Представить для проверки результат выполнения работы в виде одного или нескольких файлов с исходным кодом на языке программирования PHP.

Стандартное задание

Вариант 1: объявить переменную каждого из поддерживаемых PHP типов данных, исследовать поведение PHP при выполнении всех видов арифметических операций со всеми объявленными переменными.

Вариант 2: объявить переменную каждого из поддерживаемых PHP типов данных, вывести переменные на экран тремя разными способами.

Вариант 3: сгенерировать HTML-таблицу с 1000-ю строками, в которой каждая 5-я строка имеет синий фон.

Вариант 4: сгенерировать HTML-таблицу с 1000-ю строками, в которой цвет фона строк меняется от 000000 до FFFFFF одновременным увеличением всех компонент цвета на один на каждой следующей строке.

Вариант 5: в произвольном тексте каждое третье слово перевести в верхний регистр, каждую третью букву всех слов сделать фиолетовой, подсчитать общее количество встречающихся в тексте букв «о» и «О».

Вариант 6: написать скрипт, получающий в качестве параметра командной строки десятичное число и представляющий его в системах счисления от двоичной до 16-ричной.

Вариант 7: объявить пятимерный массив с произвольными данными (не менее 30-ти элементов), вывести массив на экран таким образом, чтобы элементы первого уровня отображались красным цветов, второго – синим, третьего – зелёным, четвёртого – фиолетовым, пятого – жёлтым.

Вариант 8: объявить пятимерный массив с произвольными данными (не менее 30-ти элементов), в этом массиве (программно!) удалить все целые числа, дроби округлить до сотых, все текстовые элементы перевести в верхний регистр.

Вариант 9: объявить пятимерный массив с произвольными данными (не менее 30-ти элементов), в этом массиве (программно!) удалить целые числа, дроби округлить до сотых, текстовые элементы перевести в верхний регистр.

Вариант 10: объявить пятимерный массив с произвольными данными (не менее 30-ти элементов), в этом массиве (программно!) отсортировать все данные по возрастанию в строковом режиме, а также подсчитать (программно!) количество числовых элементов.

Расширенное задание

Написать элементарный шаблонизатор, выполняющий поиск и подстановку в шаблонах элементов без параметров (например {TIME}, {DATE} и тому подобное). В качестве шаблонов использовать результат расширенного задания первой лабораторной работы.

Результатом данной работы должен стать скрипт, генерирующий динамически часть содержимого страниц сайта.

Логика работы скрипта такова: определить запрашиваемую страницу, прочитать с диска её исходный код, провести подстановку элементов, отобразить результат.

Если вам хватает знаний, вы можете сразу переходить к разработке более универсального механизма обработки шаблонов – фактически к написанию собственного шаблонизатора, который мог бы выполнять такие действия, как сборка шаблона из подшаблонов, подстановка данных из конфигурационных файлов, базы данных и динамически сформированных значений переменных (массивов).

Рекомендации по выполнению заданий

В данной лабораторной работе особое внимание следует уделить оформлению кода (отступы, комментарии, наименования переменных и функций), а также вопросам универсальности алгоритма.

Разработанный алгоритм должен быть независимым от данных, т. е. продолжать корректно функционировать, если входные данные будут заменены на иной произвольный набор значений – как корректный, так и некорректный.

Таким образом, в данной лабораторной работе мы постепенно приближаемся к вопросам обеспечения качества приложений в контексте устойчивости к входным данным.

При именовании переменных и функций, а также при оформлении кода рекомендуется придерживаться следующих правил:

- все имена пишутся в одном стиле;
- имена переменных пишутся в нижнем регистре, состоят из не более чем 2–3 слов, разделённых знаком подчёркивания и представляющих собой существительные или прилагательные;
- имена функций пишутся в нижнем регистре, состоят из не более чем 2–3 слов, разделённых знаком подчёркивания и представляющих собой глаголы или существительные;
- имена переменных и функций являются мнемоничными (отражают смысл хранимых данных или выполняемых действий);
- рекомендуемое количество комментариев – одна строка на 3–5 строк кода программы;
- отступы оформляются знаком табуляции или пятью пробелами;
- варианты поведения программы в условных конструкциях заключаются в операторные скобки даже тогда, когда состоят из одного оператора.

Лабораторная работа №3. Специальные функции PHP

Цель работы: изучение функций, определяемых пользователем, функций по работе с датой и временем, функций по работе с файловой системой языка программирования PHP.

Порядок выполнения работы

1. Изучить темы 3.8–3.10 лекционного материала.
2. Выполнить задание по лабораторной работе.
3. Представить для проверки результат выполнения работы в виде одного или нескольких файлов с исходным кодом на языке программирования PHP.

Стандартное задание

Вариант 1: написать функцию, определяющую точный возраст человека (с точностью до одного дня) по его дате рождения. Дату рождения получать через веб-форму.

Вариант 2: написать функцию, формирующую полный список файлов в указанном каталоге (включая подкаталоги) и считающую общий объём файлов. Имя каталога, в котором следует выполнять поиск, получать через веб-форму.

Вариант 3: написать функцию, формирующую календарь на год. Календарь представить в виде HTML-таблицы. Год, за который следует формировать календарь, получать через веб-форму.

Вариант 4: написать функцию, получающую имя файла и приводящую его в соответствие со следующими правилами: допустимы только английские буквы в нижнем регистре, цифры, знаки подчёркивания и не более одной точки; русские буквы транслитерировать в английские; остальные недопустимые символы заменить на знаки подчёркивания; если такой файл существует в указанном каталоге, добавлять в конец имени (перед расширением) постфикс "_1", "_2" и так далее до получения уникального имени файла. Имя файла и каталога получать через веб-форму.

Вариант 5: написать функцию, формирующую список файлов в указанном каталоге (включая подкаталоги), время создания которых лежит в указанном диапазоне, а имя содержит указанное сочетание символов. Данные для поиска получать через веб-форму.

Вариант 6: написать функцию, выполняющую поиск указанного значения в произвольном массиве. Результат поиска представлять в виде массива, элементами которого являются массивы, содержащие путь к найденным вхождениям элемента в исходном массиве.

Вариант 7: написать функцию, определяющую процентное отношение объёма графических файлов в произвольном каталоге (включая подкаталоги) к общему объёму данных в этом каталоге. Имя каталога получать через веб-форму.

Вариант 8: написать функцию, формирующую полный список файлов и подкаталогов в указанном каталоге. Для всех элементов списка выводить размер в килобайтах (для подкаталогов считать размер их содержимого), дату и время создания, модификации и последнего обращения. Для всех текстовых файлов отобразить первые 100 символов. Имя анализируемого каталога получать через веб-форму.

Вариант 9: написать функцию, формирующую календарь учебного года с указанием номера учебной недели. Первой неделей учебного года считается неделя, на которую приходится 1-е сентября. Номера учебных недель – от 1-го до 4-х. Год, для которого следует формировать календарь учебных недель, получать через веб-форму.

Вариант 10: написать функцию, преобразующую число в словесную форму записи (например 127 преобразуется в "сто двадцать семь"). Число (до 20 разрядов) получать через веб-форму.

Расширенное задание

Написать часть системы управления сайтом, отвечающую за добавление, удаление и перемещение файлов. Фактически требуется реализовать примитивный веб-ориентированный файловый менеджер для управления файловой системой на стороне сервера.

Дополнительными возможностями такого файлового менеджера может быть управление каталогами, редактирование текстовых файлов, конвертация графических файлов, шифрование и дешифрование файлов, создание и распаковка архивов и т. п.

Рекомендации по выполнению заданий

Взаимодействие с файловой системой и работа с датами – операции, при выполнении которых пользователи часто совершают ошибки. Ваша задача – реализовать полученное задание настолько устойчивым к нестандартным ситуациям, насколько это возможно.

Так, например, должны быть проверки на корректность и соответствие здравому смыслу введённых дат, на существование и тип объектов файловой системы, с которыми пользователь собирается выполнять операции и т. п.

В случае возникновения нештатной ситуации ваша программа должна в максимально удобной для пользователя форме реагировать на происходящее, предлагая варианты решения и предотвращая необратимые действия пользователя, которые могут привести к повреждению или потере данных.

Для всех программ, в которых данные запрашиваются через веб-форму, в случае некорректного ввода данных веб-форма должна отображаться вновь. При этом все поля должны сохранить введённые пользователем значения, а неверно заполненные поля должны быть отмечены красным цветом и дополнены подсказкой, поясняющей, в чём суть допущенной пользователем ошибки ввода.

Лабораторная работа №4. Регулярные выражения в PHP

Цель работы: изучение основ регулярных выражений и их использования в языке программирования PHP.

Порядок выполнения работы

1. Изучить тему 3.11 лекционного материала.

2. Выполнить задание по лабораторной работе (во всех вариантах использование регулярных выражений является ОБЯЗАТЕЛЬНЫМ!)

3. Представить для проверки результат выполнения работы в виде одного или нескольких файлов с исходным кодом на языке программирования PHP.

Стандартное задание

Вариант 1: в произвольном тексте все целые числа вывести синим цветом, все дроби вывести красным цветом и округлить до десятых.

Вариант 2: в произвольном тексте все аббревиатуры вывести красным цветом, все слова, начинающиеся с большой буквы, вывести зелёным цветом, все числа подчеркнуть.

Вариант 3: в произвольном тексте все e-mail адреса вывести красным цветом и привести к виду EMAIL.

Вариант 4: в произвольном тексте все URL'ы вывести красным цветом и привести к виду URL. Если до преобразования присутствовала человекочитаемая часть URL'а, выводить URL в виде URL; человекочитаемая_часть.

Вариант 5: в произвольном тексте все номера телефонов (предусмотреть не менее пяти вариантов записи номера) вывести зелёным цветом. При этом номера сотовых телефонов (начинаются с "+КОД-") подчеркнуть.

Вариант 6: в произвольном тексте все даты (в формате DD.MM.YYYY и MM/DD/YYYY, причём день и месяц могут быть однозначными, а год – двузначным) вывести красным цветом, при этом увеличить год на единицу.

Вариант 7: в произвольном тексте все слова, состоящие из английских букв, вывести синим цветом, все слова, состоящие из русских букв, вывести красным цветом, все числа вывести зелёным цветом.

Вариант 8: в произвольном тексте последовательности из двух и более пробельных символов заменить на один пробел, каждое предложение оформить в виде отдельного абзаца, все аббревиатуры подчеркнуть, все числа вывести синим цветом.

Вариант 9: в произвольном тексте все слова, начинающиеся с большой буквы, но не стоящие в начале предложения, вывести красным цветом, а все слова, стоящие в начале предложения, подчеркнуть.

Вариант 10: в произвольном HTML-документе все подчёркнутые фрагменты текста вывести синим, все наклонные фрагменты текста вывести зелёным, все жирные фрагменты текста вывести красным цветом.

Расширенное задание

Написать шаблонизатор (программу, управляющую сборкой готовых HTML-страниц из отдельных шаблонов). Шаблонизатор должен уметь обрабатывать следующие инструкции:

{FILE="path_to_file"} – чтение и подстановка указанного файла;

{CONFIG="value"} – чтение и подстановка значения из конфигурационного файла;

{VAR="variable_name"} – подстановка значения из массива $VARS, формируемого в процессе работы приложения;

{DB="value"} – подстановка значения из предопределённой таблицы в БД, хранящей текстовые надписи, настройки приложения и тому подобную информацию;

{IF "var_1"</>/==/!=/<=/>="var2"} PART1 {ELSE} PART2 {ENDIF} – анализ условия и удаление из шаблона той части, которая не соответствует условию; условия могут быть вложенными; часть {ELSE} может отсутствовать.

Рекомендации по выполнению заданий

Основное требование при выполнении данной работы – универсальность и ориентированность на производительность.

Не допускается выполнение данной лабораторной работы в виде, когда программа реагирует лишь на некоторые частные случаи вхождения искомых элементов в текст. Следует помнить, что отдельные части искомых элементов могут быть разделены переносами строк, пробелами и иными символами, которые автор документа мог использовать для форматирования текста.

Производительность скриптов следует анализировать, разрабатывая несколько альтернативных вариантов и выбирая наиболее быстродействующий.

В некоторых случаях алгоритмически более простым является выполнение задачи в несколько этапов с предварительным приведением данных к формату, позволяющему использовать более простые регулярные выражения, нежели в случае анализа исходного текста без предобработки.

Ещё одним важным показателем качества программы является использование оперативной памяти. Поскольку вложенные регулярные выражения приводят к рекурсивным вызовам и геометрической прогрессии занимаемого анализируемыми данными объёма оперативной памяти, следует предусмотреть особое поведение программы для случаев, когда объём входных данных приближается к объёму доступной программы оперативной памяти.

В предложенных выше вариантах заданий нет строгого требования к анализу текста в различных кодировках, однако при выполнении задания рекомендуется предусмотреть представление входных данных в кодировке UTF8.

Во всех вариантах заданий результатом работы программы должна являться корректная HTML-страница, содержащая как исходный, так и преобразованный согласно заданию текст.

Лабораторная работа №5. Взаимодействие PHP с реляционными СУБД

Цель работы: изучение возможностей языка программирования PHP по взаимодействию с реляционной СУБД MySQL.

Порядок выполнения работы

1. Изучить темы 4.1–4.4 лекционного материала.
2. Выполнить задание по лабораторной работе.
3. Представить для проверки результат выполнения работы в виде одного или нескольких файлов с исходным кодом на языке программирования PHP, а также дамп использованной при выполнении задания базы данных.

Стандартное задание

Вариант 1: написать скрипт, отображающий содержимое некоторой заданной таблицы из БД, добавление, редактирование и удаление записей.

Вариант 2: написать скрипт, отображающий структуру и данные всех таблиц указанной БД.

Вариант 3: написать скрипт, формирующий в виде HTML-страницы календарь, в котором ссылками являются даты, за которые в некоторой таблице, содержащей новости, присутствуют новости.

Вариант 4: написать скрипт, выводящий в случайном порядке заданное количество неповторяющихся записей из произвольной таблицы БД.

Вариант 5: написать скрипт, позволяющий добавить в произвольную таблицу БД произвольное количество записей со случайными данными. Скрипт должен получать в качестве входных данных имена и типы («число» или «текст») полей таблицы, а также количество добавляемых записей.

Вариант 6: написать скрипт, получающий через форму e-mail пользователя, проверяющий его корректность и добавляющий его в таблицу БД в случае, если такого e-mail там ещё нет.

Вариант 7: написать скрипт, получающий через форму имя пользователя, пароль и подтверждение пароля. Если такого пользователя ещё нет в БД, а пароль и его подтверждение совпадают, необходимо добавить в БД имя пользователя и пароль в виде хэша sha1.

Вариант 8: сформировать БД, содержащую информацию о студентах и их оценках по различным предметам. Написать скрипт, формирующий список студентов с их средними баллами, а также минимальным и максимальным баллом с указанием списка предметов, за которые был получен такой балл.

Вариант 9: написать скрипт, позволяющий выполнить произвольный запрос к СУБД, после чего отображающий результат выполнения запроса и статистику: время выполнения, использованная оперативная память.

Вариант 10: написать скрипт, выполняющий указанное количество раз произвольный SQL-запрос и собирающий статистику производительности СУБД при выполнении этого запроса.

Расширенное задание

На основе результатов выполнения расширенных заданий в лабораторных работах 1–4 доработать программу формирования пользовательской и администраторской частей сайта в контексте взаимодействия с СУБД, а именно:

– реализовать хранение структуры сайта в БД;
– реализовать построение карты сайта и поиска по сайту;
– реализовать протоколирование действий администратора;
– реализовать такие модули сайта, как (на выбор) голосование, показ случайного банера, подписка на рассылку.

Рекомендации по выполнению заданий

Одной из наиболее часто встречающихся проблем при работе с базами данных является синхронизация кодировок. Чтобы избежать данной проблемы, необходимо следовать правилу: кодировки должны быть одинаковыми в:

– среде разработки приложения;
– файлах шаблонов HTML-страниц;
– конфигурационных файлах;
– таблицах БД;
– настройках взаимодействия MySQL и PHP.

Поскольку UTF8 становится стандартом де-факто, предлагается использовать именно эту кодировку, а в целях устранения проблем после установления соединения с СУБД и выбора БД следует выполнить два запроса:

`SET CHARACTER SET 'UTF8'` и `SET NAMES 'UTF8'`

Данная лабораторная работа не подразумевает разработку сложных SQL-запросов, однако следует уделить внимание корректности работы с СУБД на стороне PHP.

Для взаимодействия с MySQL в PHP предусмотрено три стандартных решения: использование расширения mysql – классический вариант, постепенно начинающий устаревать, использование расширения mysqli (MySQL Improved) – наиболее активно развивающийся способ, рекомендованный ныне к использованию большинством специалистов; использование расширения PDO (PHP Data Objects) – одно из наиболее перспективных направлений развития взаимодействия PHP с реляционными СУБД, предоставляющее дополнительный уровень абстракции и повышающий таким образом совместимость написанных на PHP программ с различными СУБД.

В данной лабораторной работе не предъявляется жёстких требований к использованию того или иного варианта взаимодействия PHP и MySQL, однако особо рекомендуется (прежде всего – в расширенном задании) хотя бы попробовать все три варианта, причём основной акцент сделать на применение расширения mysqli.

Лабораторная работа №6. Принципы разделения дизайна и кода

Цель работы: изучение принципов разделения дизайна и кода, технологий формирования ядра программных средств, отвечающих за функционирование интернет-ориентированных приложений.

Порядок выполнения работы

1. Изучить темы 5.3–5.5 лекционного материала.
2. Выполнить задание по лабораторной работе.
3. Представить для проверки результат выполнения работы в виде одного или нескольких файлов с исходным кодом на языке программирования PHP, а также файлов, содержащих HTML-шаблоны.

Стандартное задание

Вариант 1: написать скрипт, формирующий и обрабатывающий HTML-форму регистрации пользователя на сайте. Для формирования формы использовать шаблоны.

Вариант 2: написать скрипт, формирующий страницу сайта на основе не менее чем пяти отдельных фрагментов HTML-шаблонов с произвольными именами.

Вариант 3: написать скрипт, формирующий и обрабатывающий HTML-форму оформления заказа товара. Для формирования формы использовать шаблоны.

Вариант 4: написать скрипт, формирующий и обрабатывающий блок опроса мнения посетителей сайта. Для формирования блока использовать шаблоны.

Вариант 5: написать скрипт, формирующий список новостей. Для отображения отдельной новости и формирования всей страницы использовать шаблоны.

Вариант 6: написать скрипт, формирующий список товаров интернет-магазина. Для отображения отдельного товара и формирования всей страницы использовать шаблоны.

Вариант 7: написать скрипт, формирующий блок банерной рекламы. Для отображения отдельного банера и формирования всей страницы использовать шаблоны.

Вариант 8: написать скрипт, формирующий блок с прогнозом погоды на ближайшие три дня. Для отображения всего прогноза, отдельного дня в прогнозе и формирования всей страницы использовать шаблоны.

Вариант 9: написать скрипт, формирующий блок с курсами валют за вчера и сегодня. Для отображения всего блока, отдельного дня и формирования всей страницы использовать шаблоны.

Вариант 10: написать скрипт, генерирующий веб-форму на основе набора полей и отдельных шаблонов для всей формы и каждого поля.

Расширенное задание

На основе выполнения лабораторных работ 4–5 расширить возможности создаваемого программного средства, добавив поддержку стилевого оформления внешнего вида сайта. Под «стилем » в данном случае понимается отдельный набор HTML-шаблонов, CSS-файлов, графических элементов, подключаемый администратором ресурса с целью быстрого изменения внешнего вида ресурса при сохранении имеющегося набора функций.

Дополнительным заданием, выполнение которого позволит намного глубже изучить вопросы шаблонизации, является доработка результатов лабораторной работы 4 добавлением функции кэширования как результатов сборки всей страницы из шаблонов, так и результатов частичной сборки страницы.

Такой подход позволяет значительно ускорить генерацию страниц, большая часть которых остаётся неизменной при большинстве запросов пользователей, в то время как некоторые части меняются при каждом запросе.

Рекомендации по выполнению заданий

Основные правила, которые следует уяснить при выполнении данной лабораторной работы:

- HTML-шаблоны формируются *человеком*;
- файлы HTML-шаблонов *не изменяются* в процессе формирования страниц: изменения происходят *только* в оперативной памяти;
- внутри HTML-шаблонов *не должно быть никакой человеко-ориентированной* информации (никакого текста, надписей и тому подобного) – допускается *только* HTML/CSS-код;
- HTML-шаблоны и логику их обработки следует проектировать с учётом того факта, что *отдельные элементы контента могут как присутствовать, так и отсутствовать* в процессе эксплуатации интернет-ресурса (так, у новости может не быть сопутствующей фотографии, у пользователя может не быть подписи и тому подобное);
- правильный подход к проектированию системы управления шаблонами позволяет *полностью исключить* наличие PHP-кода внутри HTML-документов и HTML/CSS-кода внутри скриптов PHP;
- правильная система управления шаблонами является *универсальной*, т. е. не требует изменения своего кода при добавлении в шаблон новых плейсхолдеров поддерживаемых типов.

Поскольку процесс сборки HTML-страницы из шаблонов и наполнения её информацией является вычислительно ёмкой задачей, при выполнении данной лабораторной работы следует изучить возможность кэширования конечных или промежуточных результатов сборки HTML-страницы.

Глубокое понимание данной темы составляет примерно треть необходимых для разработки веб-приложений навыков: подойдите к работе ответственно.

Лабораторная работа №7. Сессии и куки в PHP

Цель работы: изучение механизмов управления сессиями и куки в языке программирования PHP.

Порядок выполнения работы

1. Изучить тему 6.6 лекционного материала.
2. Выполнить задание по лабораторной работе.
3. Представить для проверки результат выполнения работы в виде одного или нескольких файлов с исходным кодом на языке программирования PHP.

Стандартное задание

Вариант 1: написать два скрипта, один из которых формирует произвольный набор данных (числа, строки, массивы) и передаёт их другому скрипту в сериализованной форме. Второй скрипт десериализует данные.

Вариант 2: написать скрипт, позволяющий установить пользователю произвольную куки (с произвольным именем и значением, на произвольный срок), просмотреть список установленных куки.

Вариант 3: написать скрипт, позволяющий установить пользователю произвольную куки (с произвольным именем и значением, на произвольный срок) и удалить её.

Вариант 4: написать скрипт, позволяющий просматривать и редактировать список установленных у пользователя куки.

Вариант 5: написать скрипт, выполняющий авторизацию пользователя на сайте.

Вариант 6: написать скрипт, выполняющий авторизацию пользователя на сайте с возможностью долговременной авторизации через куки (функция «запомнить меня»).

Вариант 7: написать скрипт, позволяющий определять, какие страницы сайта посетил пользователь (при этом пользователь не регистрируется и не авторизуется на сайте).

Вариант 8: написать скрипт, позволяющий определять, когда данный пользователь посещал сайт (полный список посещений). При этом пользователь не регистрируется и не авторизуется на сайте.

Вариант 9: написать скрипт, позволяющий определять «путь пользователя по сайту» (какие страницы он посетил, в какой последовательности, в какие моменты времени). При этом пользователь не регистрируется и не авторизуется на сайте.

Вариант 10: написать скрипт, реализующий «электронную корзину» в интернет-магазине. Корзина должна хранить перечень и количество выбранных пользователем товаров. При этом пользователь не регистрируется и не авторизуется на сайте, а после подтверждения пользователем заказа корзина автоматически очищается.

Расширенное задание

В программном средстве, которое получилось в результате выполнения лабораторных работ 1–6, реализовать механизм регистрации и авторизации пользователей, механизм авторизации администратора, механизм управления списком и набором прав пользователей.

В механизме авторизации пользователей предусмотреть возможность долговременной авторизации (функция «запомнить меня»), а также кратковременной авторизации с максимальной защитой личных данных (функция «чужой компьютер»).

В случае, если разрабатываемое вами программное средство допускает использование электронной корзины, реализовать её.

В случае, если разрабатываемое вами программное средство допускает использование поиска по сайту, доработать его таким образом, чтобы для каждого посетителя (в т.ч. такого, который не зарегистрирован и не авторизован) хранилась история его поисковых запросов.

Рекомендации по выполнению заданий

Механизм сессий и куки – один из немногих способов предоставить веб-серверу возможность «опознавать» запросы, как приходящие от одного и того же пользователя.

Суть использования куки заключается в сохранении браузером данных на стороне клиента и отправке их на сторону сервера при каждом запросе.

Суть использования сессий заключается в передаче на сторону клиента специального идентификатора (который может храниться в куки или передаваться через ссылки и формы), с которым проассоциирован набор данных, сохранённый на стороне сервера.

Использование куки позволяет посетителям ресурсов применять некоторые настройки без регистрации, а серверу позволяет опознавать таких пользователей и применять их настройки, а также отслеживать статистику использования этими пользователями ресурса.

С использованием сессий и куки связано некоторое количество потенциальных проблем в области безопасности:

- куки могут быть украдены злоумышленником, а потому не следует хранить в них конфиденциальные данные;
- куки могут быть модифицированы пользователем, а потому данные, полученные из них, следует подвергать тщательной фильтрации наряду с данными, полученными через формы или ссылки;
- сессии являются более безопасным способом хранения информации, т. к. она находится только на стороне сервера, однако атака типа «кража сессии» (кража и подмена идентификатора сессии) позволяет злоумышленнику достичь своей цели и в такой ситуации, что требует принятия дополнительных мер безопасности.

Лабораторная работа №8. Генерация и анализ статистики, работа с почтой в PHP

Цель работы: изучение технологий генерации и анализа статистики использования интернет-ресурсов, изучение технологий работы с почтовыми сообщениями с помощью языка программирования PHP.

Порядок выполнения работы

1. Изучить темы 7.1–7.2 и 8.1–8.2 лекционного материала.
2. Выполнить задание по лабораторной работе.
3. Представить для проверки результат выполнения работы в виде одного или нескольких файлов с исходным кодом на языке программирования PHP.

Стандартное задание

Вариант 1: написать скрипт, собирающий статистику по используемым посетителями ресурса браузерам. Выводить результаты в виде HTML-таблицы со списком браузеров, отсортированным по убыванию количества пользующихся ими посетителей сайта.

Вариант 2: написать скрипт, собирающий статистику по используемым посетителями ресурса операционным системам. Выводить результаты в виде HTML-таблицы со списком операционных систем, отсортированным по убыванию количества пользующихся ими посетителей сайта.

Вариант 3: написать скрипт, собирающий статистику по времени посещения сайта. Выводить результаты в виде графиков активности посетителей за день, неделю, месяц, год. График представить в виде HTML.

Вариант 4: написать скрипт, собирающий статистику по ip-адресам, с которых посетители заходили на сайт. Выводить результаты в виде HTML-таблицы со списком ip-аресов, отсортированным по убыванию количества посещений с каждого адреса.

Вариант 5: написать скрипт, отправляющий полученное через форму письмо списку адресатов, хранящемуся в БД.

Вариант 6: написать скрипт, отправляющий полученное через форму письмо указанному адресату.

Вариант 7: написать скрипт, отправляющий полученное через форму письмо указанному адресату. Письмо может содержать произвольное количество вложений (attachments).

Вариант 8: написать скрипт, отправляющий администратору статистику посещения ресурса за день (название страницы, количество просмотров).

Вариант 9: написать скрипт, осуществляющий подписку пользователя на рассылку. Подписка производится после подтверждения, информация о котором выслана на указанный пользователем e-mail.

Вариант 10: написать скрипт, формирующий и обрабатывающий «форму обратной связи». Пользователь может выбрать адресата сообщения из списка.

Расширенное задание

В случае добросовестного выполнения расширенных заданий к предыдущим лабораторным работам к данному моменту у вас должно получиться вполне работоспособное программное средство управления структурой и информационным наполнением интернет-ресурса (CMS – content management system).

Теперь остаётся лишь добавить в это программное средство модуль, отвечающий за сбор и анализ статистики и осуществляющий рассылку корреспонденции.

Задание на данную лабораторную работу простое:

- доработать (создать) модуль подписки пользователей на рассылку уведомлений и модуль рассылки таких уведомлений;
- создать модуль сбора и анализа статистики посещения сайта;
- создать модуль отправки администратору сайта отчётов о статистике посещений сайта и возникших нештатных ситуациях.

Рекомендации по выполнению заданий

При сборе и анализе статистики в общем случае учитывается такая информация о посетителе ресурса: ip-адрес, браузер и операционная система, страна, откуда зашёл посетитель (можно определить по ip), домен и конкретный адрес страницы, с которой посетитель пришёл на сайт, как долго посетитель находился на той или иной странице (т. е. время между запросами страниц), путь пользователя по сайту (удобно для оптимизации навигации), клики пользователя по банерам (можно включать в общую статистику).

Также учитывается количество уникальных посетителей в час, день, неделю, месяц и т.д., полученный и переданный объём данных, какие поисковые системы и как часто индексируют сайт, по каким ключевым словам посетители находят сайт в поисковых системах и тому подобное.

Важным вопросом в контексте сбора и анализа статистики является производительность: процесс агрегации данных может занимать значительное время, а потому стоит реализовать отдельные функции для обработки оперативных данных и агрегации данных.

Представление итоговых результатов удобно реализовывать с помощью графических изображений (например в формате PNG). Информацию о работе с изображениями вы можете найти в теме 7.2 лекционного материала.

Отправка почты большому количеству адресатов сопряжена с риском исчерпать лимит времени, отводимый PHP на выполнение скрипта. В связи с этим рекомендуется дополнительно изучить способы разбиения объёмных задач на подзадачи, которые будут выполняться при последующих запусках скрипта, а также способы создания «бессмертных» скриптов, которые при достижении лимита времени запускают свою новую копию, выполняя HTTP-запрос к серверу.

Примеры выполнения лабораторных работ

Пример выполнения работы №1

Вариант N: разработать веб-страницу (HTML, CSS), содержащую форму, представленную на рисунке 5. Использовать табличную вёрстку.

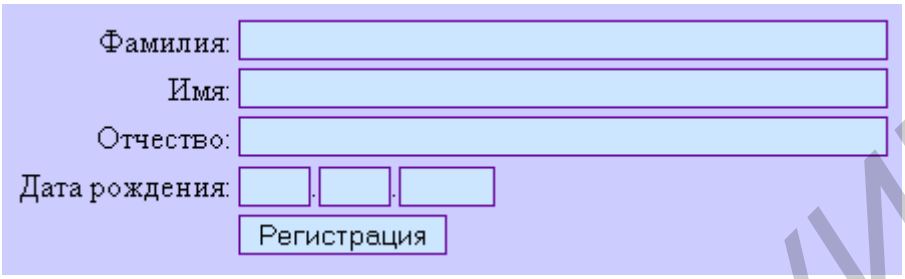

Рисунок 5 – Форма для варианта N

Решение

Файл form.html

```
<html>
 <head>
  <title>Регистрация пользователя</title>
  <meta http-equiv="Content-Type" content="text/html;
                                          charset=windows-1251" />
  <link rel="stylesheet" type="text/css" media="screen"
                                          href="form.css" />
 </head>

 <body>

 <table border="0">
 <form action="#" method="post">
  <tr>
   <td align="right">Фамилия:</td>
   <td align="left"><input type="text" name="name_l" size="50"
                                  maxlength="200" value="" /></td>
  </tr>
  <tr>
   <td align="right">Имя:</td>
   <td align="left"><input type="text" name="name_f" size="50"
                                  maxlength="200" value="" /></td>
  </tr>
  <tr>
   <td align="right">Отчество:</td>
   <td align="left"><input type="text" name="name_m" size="50"
                                  maxlength="20" value="" /></td>
  </tr>
  <tr>
   <td align="right">Дата рождения:</td>
   <td align="left">
     <input type="text" name="bd_d" size="2" maxlength="2" value="" />.
```

```
     <input type="text" name="bd_m" size="2" maxlength="2" value="" />.
     <input type="text" name="bd_y" size="4" maxlength="4" value="" />
</td>
  </tr>
  <tr>
   <td align="right"> </td>
   <td align="left"><input type="submit" name="go"
                                    value="Регистрация" /></td>
  </tr>
 </form>
 </table>

 </body>
</html>
```

Файл form.css

```
body {background-color: #CCCCFF}
input {border: 1px solid #660099; background-color: #CCE6FF}
```

Пример выполнения работы №2

Вариант N: написать скрипт, получающий в качестве параметра командной строки двоичное число и представляющий его в десятичной системе счисления.

Решение

Файл lab2.php

```
<?php

 // Если из командной строки не передано число -- прекратить работу.
 if ($argc<2)
     {
          die("Укажите двоичное число в качестве параметра командной
                                                          строки.");
     }

 // Извлечение числа из параметров, переданных из командной строки.
 $num = $argv[1];

 // Проверка длины введ□нных данных.
 if (strlen($num)>100)
     {
          die('Вы ввели слишком длинное число.');
     }

 /* Проверка того, что из командной строки передано двоичное число.
    Данную проверку можно выполнить более эффективно с помощью регулярных
    выражений, что и будет показано в закомментированном блоке кода. */
```

```
for ($i=0; $i<strlen($num); $i++)
     {
          if (($num[$i]!='0')&&($num[$i]!='1'))
               {
                    die('Переданное число не является двоичным.');
               }
     }

/* Та же проверка через регулярные выражения:
if (preg_match("/[^01]/", $num))
     {
          die('Переданное число не является двоичным.');
     }
*/

// Конвертация и вывод результата.
echo bindec($num);

?>
```

Пример выполнения работы №3

Вариант N: написать функцию, формирующую список только файлов в указанном из командной строки каталоге. В список включить размер файлов в байтах и килобайтах, а также время создания, модификации и последнего обращения к файлу.

Решение

Файл lab3.php

```
<?php

 // Если из командной строки не передано имя
 // каталога -- прекратить работу.
 if ($argc<2)
      {
           die("Укажите имя каталога в качестве параметра командной
                                                              строки.");
      }

 // Извлечение имени каталога из
 // параметров, переданных из командной строки.
 $dir_name = $argv[1];

 // Проверка существования каталога.
 if (!is_dir($dir_name))
      {
           die('Такого каталога не существует.');
      }

 // Открытие каталога.
 $dir_resource = opendir($dir_name);

 // Проверка успешности операции открытия каталога.
```

```
if ($dir_resource === NULL)
     {
          die('Не удалось открыть каталог.');
     }

// Чтение каталога.
while (($file_name = readdir($dir_resource)) !== false)
     {
          // Получение полного имени элемента.
          $full_name = $dir_name.'/'.$file_name;

          // Проверка того, что элемент
          // является файлом.
          if (is_file($full_name))
               {
                    // Вывод информации.
                    echo $file_name.' ';
                    echo filesize($full_name).
          ' ('.round(filesize($full_name)/1024,2).' Kb) ';
                    echo date('Y.m.d H:i:s',
                              filectime($full_name)).', ';
                    echo date('Y.m.d H:i:s',
                              filemtime($full_name)).', ';
                    echo date('Y.m.d H:i:s',
                              fileatime($full_name))."\n";
               }
     }

// Закрытие каталога.
closedir($dir_resource);

?>
```

Пример выполнения работы №4

Вариант N: в произвольном тексте все целые числа возвести в квадрат и вывести красным цветом.

Решение

Файл lab4.php

```
<?php

 // Чтение текста из файла.
 if (is_file('text.txt'))
     {
          $text = file_get_contents('text.txt');
     }
     else
     {
          die('Файл данных text.txt не найден.');
     }

 // Callback-функция возведения в квадрат
```

```
 // найденных целых чисел.
 function square($x)
      {
           return '<span style="color:red">'.($x[0]*$x[0]).'</span>';
      }

 // Поиск и замена целых чисел.
 $text = preg_replace_callback('/\d+/', 'square', $text);

 // Вывод результата
 echo $text;

?>
```

Пример выполнения работы №5

Вариант N: написать скрипт, выполняющий указанное количество раз случайный запрос (из некоторого набора) и определяющий минимальное, максимальное и среднее время выполнения запроса.

Решение

Файл lab5.php

```
<?php

// Количество повторений запроса
$iterations = 100;

// Массив с запросами
$queries = array (
 "show databases",
 "show tables from 'mysql'",
 "select * from 'mysql'.'mysql_user'",
 "select * from 'mysql'.'help_relation'",
 "select * from 'mysql'.'help_topic'");

// Подсоединение к СУБД
$lnk = mysql_connect('127.0.0.1', 'root', '123456');
if ($lnk === FALSE) { die(); }

// инициализация массива оставшихся итераций для каждого запроса
for ($i=0;$i<sizeof($queries);$i++)
 {
  $final_data[$i]['iterations'] = $iterations;
 }

// Продолжать выполнение следует в случае, если хотя бы
// для одного запроса не был исчерпан лимит итераций
function can_continue($final_data)
 {
  $sum = 0;
  for ($i=0;$i<sizeof($final_data);$i++)
   {
    $sum+=$final_data[$i]['iterations'];
   }
```

```
  if ($sum>0)
   {
    return TRUE;
   }
    else
   {
    return FALSE;
   }
 }

// Выполнять следует случайный запрос, однако при
// этом у него не должен быть исчерпан лимит итераций
function get_query_number($final_data)
 {
  $x = mt_rand(0, sizeof($final_data)-1);
  if ($final_data[$x]['iterations']>0)
   {
    return $x;
   }
    else
   {
    for ($i=0; $i<sizeof($final_data); $i++)
       {
        if ($final_data[$i]['iterations']>0)
         {
          return $i;
         }
       }
   }
 }

// Функция подсч□та среднего времени выполнения запроса
function avg($arr)
 {
  if (sizeof($arr)==0)
   {
    return 0;
   }
  $sum = 0;
  for ($i=0; $i<sizeof($arr); $i++)
   {
    $sum+=$arr[$i];
   }
  return $sum/sizeof($arr);
 }

// Основной цикл: выполнение запросов и сбор значений времени
while (can_continue($final_data))
 {
  $x = get_query_number($final_data);
  $t1 = microtime(TRUE);
  mysql_query($queries[$x]);
  $t2 = microtime(TRUE);
  $final_data[$x]['avg'][] = $t2-$t1;
  $final_data[$x]['iterations']--;
 }
```

```
// Генерация и вывод результатов
for ($i=0;$i<sizeof($queries);$i++)
 {
  echo 'Query ('.($i+1).') ['.$queries[$i].']:'."\n";
  echo 'Min: '.min($final_data[$i]['avg'])."\n";
  echo 'Max: '.max($final_data[$i]['avg'])."\n";
  echo 'Avg: '.avg($final_data[$i]['avg'])."\n\n";
 }

// Закрытие соединения с СУБД
mysql_close($lnk);
?>
```

Пример выполнения работы №6

Вариант N: написать скрипт, получающий через форму номер паспорта пользователя, проверяющий его корректность (две латинские заглавные буквы, семь цифр) и добавляющий его в таблицу БД в случае, если такого номера там ещё нет.

Решение

Файл lab6.php

```
<?php
// Функция генерации формы
function get_form($msg='', $passport='')
 {
  $t  = '<b>'.$msg.'</b><br/>';
  $t .= '<form action="'.$_SERVER['PHP_SELF'].'" method="post">';
  $t .= '<input type="text" name="passport" value="'.$passport.'" />';
  $t .= '<input type="submit" value="Go!" />';
  $t .= '</form>';
  return $t;
 }

// Если из формы не переданы данные,
// следует просто показать форму
if (!isset($_POST['passport']))
 {
  echo get_form();
 }
  else
 {
  // Проверка корректности данных
  if (!preg_match("/^[A-Z]{2}\d{7}$/", $_POST['passport']))
   {
    echo get_form('Вы неверно указали номер паспорта!', html-
specialchars($_POST['passport']));
   }
    else
   {
    // Установка соединения с СУБД и выбор БД
```

```
          $lnk = mysql_connect('127.0.0.1', 'root', '123456');
          mysql_select_db('db', $lnk);

          // Выполнение запроса
          mysql_query("INSERT into 'passports' ('passport') VALUES
('".mysql_real_escape_string($_POST['passport'],        $lnk)."')",
$lnk);

          // Анализ результата выполнения запроса
          if (mysql_affected_rows($lnk)==1)
           {
            echo get_form('Данные добавлены успешно!');
           }
            else
           {
            echo get_form('Такой номер паспорта уже есть в БД!',
htmlspecialchars($_POST['passport']));
           }

          // Закрытие соединения с СУБД
          mysql_close($lnk);
        }
      }

     ?>
```

Пример выполнения работы №7

Вариант N: написать скрипт, устанавливающий пользователю десять куки со случайными именами и значениями, срок действия первой из которых – один час, а у каждой следующей на час больше.

Решение

Файл lab7.php

```
<?php

// Функция генерации случайного имени произвольной длины
function get_random_name($len)
 {
  $name = '';

  for ($i=0; $i<$len; $i++)
   {
    $char = chr(mt_rand(ord('a'), ord('z')));
     $name .= $char;
   }

  return $name;
 }
```

```
// Срок годности первой куки
$best_before = time()+3600;

// Установка куки
for ($i=0; $i<10; $i++)
 {
  setcookie(get_random_name(10), mt_rand(0,1000), $best_before);
  $best_before+=3600;
 }


?>
```

Пример выполнения работы №8

Вариант N: написать скрипт, предоставляющий пользователю информацию, которую сервер сумел получить о его браузере, операционной системе, принимаемых кодировках, ip-адресе и иных параметрах.

Решение

Файл lab8.php

```
<?php

      echo 'Информация о вашем браузере и ОС:
'.$_SERVER['HTTP_USER_AGENT'].'<br />';
      echo 'Ваш браузер принимает документы:
'.$_SERVER['HTTP_ACCEPT'].'<br />';
      echo 'Вы предпочитаете языки:
'.$_SERVER['HTTP_ACCEPT_LANGUAGE'].'<br />';
      echo 'Ваш браузер принимает кодировки:
'.$_SERVER['HTTP_ACCEPT_CHARSET'].'<br />';
      echo 'Вы зашли с ip-адреса: '.$_SERVER['REMOTE_ADDR'].'<br />';

      ?>
```

ЛИТЕРАТУРА

1 Костарев, А. PHP 5 в подлиннике / А. Костарев, Д. Котеров. – М. : BHV, 2009.
2 Холзнер, С. PHP в примерах / С. Холзнер. – М. : Бином, 2009.
3 Фридл, Дж. Регулярные выражения / Дж. Фридл. – М. : Символ, 2008.
4 Грабер, М. SQL / М. Грабер. – М. : Лори, 2009.
5 Кинкоф, Ш. HTML / Ш. Кинкоф. – М. : НТ Пресс, 2008.

Св. план 2011, поз. 51

Учебное издание

Куликов Святослав Святославович

БАНКОВСКИЕ ИНТЕРНЕТ-ТЕХНОЛОГИИ

Методическое пособие к лабораторным работам
для студентов специальности
«Программное обеспечение информационных технологий»
всех форм обучения

Редактор Е. Н. Батурчик
Корректор А. В. Бас
Компьютерная верстка Ю. Ч. Клочкевич

Подписано в печать	Формат 60x84 1/16.	Бумага офсетная.
Гарнитура «Таймс».	Отпечатано на ризографе.	Усл. печ. л.
Уч.-изд. л.	Тираж 100 экз.	Заказ

Издатель и полиграфическое исполнение: учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0494371 от 16.03.2009. ЛП №02330/0494175 от 03.04.2009.
220013, Минск, П. Бровки, 6