Государственное бюджетное профессиональное образовательное учреждение «Дзержинский педагогический колледж»

**Методические рекомендации для студентов**

**по выполнению практических работ**

**по учебной дисциплине ОП 15. Веб-программирование**

**Дзержинск – 2017**

|  |  |
| --- | --- |
| **Одобрено на заседании ПЦК преподавателей спец. \_\_\_\_\_\_\_\_\_\_\_**  **Протокол №\_\_\_\_\_\_\_от\_\_\_\_\_\_\_\_**  **Председатель ПЦК \_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_/**  **Составитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Методические рекомендации по организации практических работ студентов составлены в соответствии с требованиями федеральных государственных образовательных стандартов среднего профессионального образования по спец. 09.02.03 Прикладная информатика (по отраслям)** |

Методические рекомендации по организации и сопровождению практических работ студентов по учебной дисциплине веб-программированию специальность 09.02.03 Прикладная информатика (по отраслям)

**Содержание**

|  |  |  |
| --- | --- | --- |
| **№** |  | **СТР** |
|  | Введение |  |
|  | Установка и запуск веб-сервера с использованием установочного пакета Denverи OpenServer | 2 |
|  | Настройка конфигураций Apache, PHP и MySQL | 2 |
|  | Вычисление значений выражений по формуле в среде программирования. | 2 |
|  | Организация вывода данных в PHP | 2 |
|  | Составление линейных программ на PHP | 2 |
|  | Условный оператор (полная, неполная форма). Вложенное ветвление | 2 |
|  | Решение текстовых задач на ветвление | 2 |
|  | Решение задач с использованием оператора Выбора | 2 |
|  | Тестирование программ | 2 |
|  | Решение задач на циклы. | 2 |
|  | Тестирование программ. Составление отчетной документации | 2 |
|  | Решение задач с использованием циклов foreach. Тестирование и оптимизация работы программ | 2 |
|  | Применение функций для решения задач | 2 |
|  | Решение задач на заполнение и вывод линейного массива | 2 |
|  | Решение задач на заполнение и вывод двумерного массива. Составление отчетной документации | 4 |
|  | Решение задач на обработку строк | 2 |
|  | Ввод данных методом GET, POST при помощи форм. Обработка GETи POST массивов | 2 |
|  | Вывод серверных переменных | 2 |
|  | Подключение скриптов при помощи include(), require() | 2 |
|  | Организация и заполнение данными сессий и кук, текстовый формат json | 2 |
|  | Организация подключения и создание запроса к БД | 2 |
|  | Извлечение данных из БД | 4 |
|  | Добавление данных в БД | 4 |
|  | Изменение данных в БД | 2 |
|  | Удаление данных из БД | 4 |

**ВВЕДЕНИЕ**

С момента своего появления web изменил компьютерный мир. Масса разрозненных компьютеров и пользователей стали объединяться для обмена информацией в виде простых HTML-документов. Такова была начальная концепция web. Взаимодействие web-сервера и клиента, с помощью браузера, позволило пользователям всего мира просматривать HTML-документы с любого сервера.

Появление динамических сайтов на основе web-приложений заложило новый этап в развитии мировой паутины, теперь web-сервер мог хранить информацию в базе данных, выполнять ее поиск и динамически формировать HTML-документы по запросам пользователей. Использование языка JavaScript позволило оживить HTML-документы.

Огромный толчок в развитии web дало появление электронной коммерции. Развитие среды web сказалось на прикладном программном обеспечении. Браузер стал неотъемлемой частью операционной системы. Будущее операционных систем связывают именно с сетью Интернет и средой web. В ближайшие годы ожидается появление операционных систем с браузером интегрированным в рабочий стол, что говорит о том, что основная работа в системе будет вестись через Интернет.

Пройдя этапы развития и становления, изменилась концепция web, появился web 2.0, в котором web-сайт структурированное хранилище информации, которой управляют пользователи. Идея наполнения контента пользователями послужила толчком для создания таких сервисов как блоги, фото и видео-галереи социальные сети и другие.

Для придания дружественного вида информации появились новые технологии пользовательского интерфейса Flash и AJAX.

Появление онлайновых сервисов, таких как электронная флешка, онлайновые офисные пакеты, задает новое направление развития среды web.

Web-среда продолжает развиваться, намечаются новые направления, разрабатывается новая концепция web 3.0, в которой будет преобладать персонализация получаемой информации. О каждом пользователе будет собираться информация в виде социального графа – интересы, увлечения, место проживания. Это позволит выдавать пользователю информацию наиболее нужную для него.

**Установка и запуск веб-сервера с использованием установочного пакета Denverи OpenServer**

*Цель***:** Установить виртуальный веб - сервер Denver

*Задачи:* **-** Научиться устанавливать Denver

- Запускать и отключать веб-сервер

- Удалять виртуальный веб-сервер

*Ход работы:*

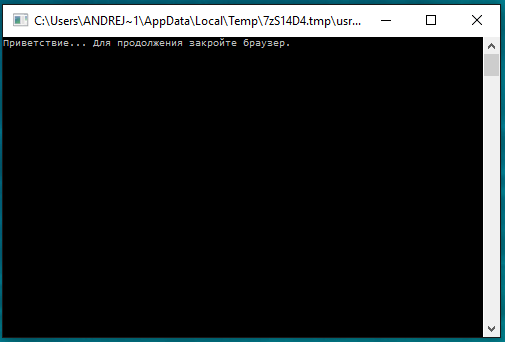
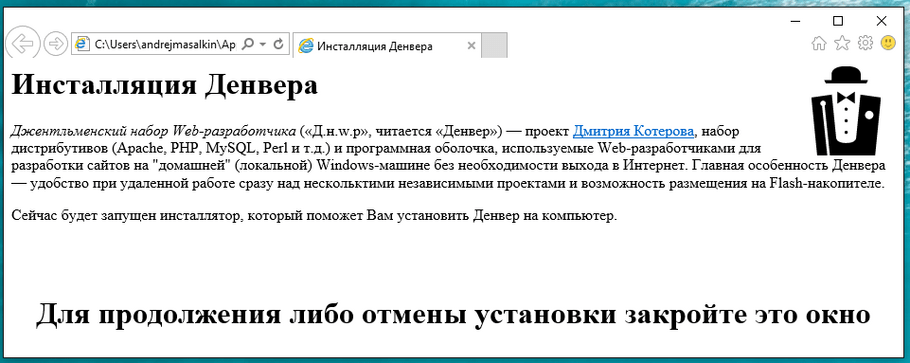
Денвер — это программный комплекс эмулирующий обычный веб-сервер системы unix. Он включает в себя сервер Apache, PHP5, MySql5, phpMyAdmin и тд. Все эти инструменты помогают развернуть на своем компьютере виртуальный хостинг и заниматься отладкой собственных сайтов без подключения к интернету. На данный момент Denwer поддерживает Windows начиная от XP и заканчивая 10 версией.

Базовый пакет Denver:

* Apache, SSL, SSI, mod*rewrite, mod*php.
* PHP5 с поддержкой GD, MySQL, sqLite.
* MySQL5 с поддержкой транзакций.
* phpMyAdmin — панель управления базой данных MySQL.
* Эмулятор sendmail и SMTP-сервера.
* Система управления виртуальными хостами.

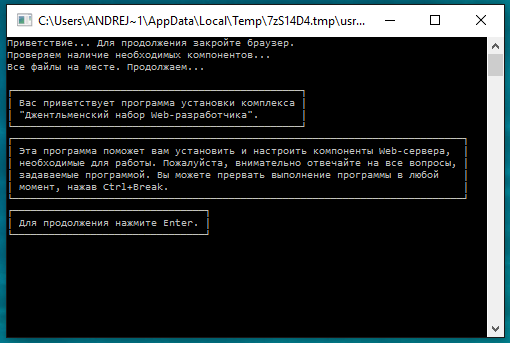
Устанавливаем Denwer

Соответственно соглашаемся. Далее ждем несколько секунд пока пакет распакуется, и видим уже вот такую картинку.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/2.png)[](https://masalkin.name/upload/image_post/create/2016/04/22/big/3.png)Продолжаем установку Денвера

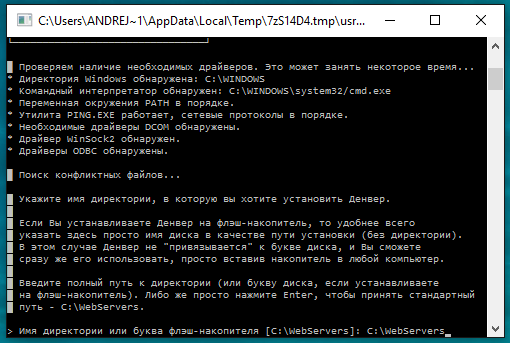
Откроется терминал с приветствием, и браузер, оповещающий нас о инсталяции Денвера.

Чтобы продолжить установку дальше, закрываем браузер. После чего терминал выдаст нам очередное приветствие, и для продолжения установки попроси нажать enter.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/4.png)

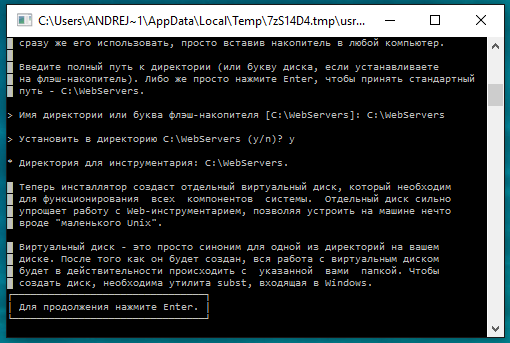
Дальше все рекомендации по установке мы будем получать в терминале. Не забыли начать Enter?

Получаем очередные рекомендации. На этом шаге нам нужно указать место, где будет размещаться виртуальный сервер. Если вы хотите установить на флешку или другой накопитель, то прописывайте путь до него. Если же собираетесь устанавливать на компьютер, то можете сделать так же как и я. Прописать пусть C:\WebServers

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/5.png)Выбираем диск куда будем устанавливать Denwer

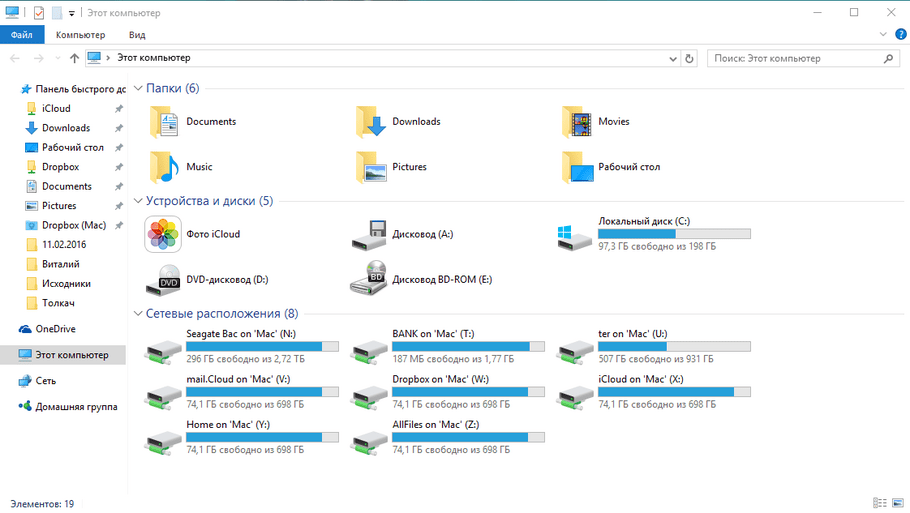
И вновь жмем Enter. Терминал вас переспросит, действительно ли вы хотите установить вебсервер в директорию C:WebServers? Ван необходимо нажать на кнопку Y, тем самым сказать yes. Не забудьте сменить раскладку клавиатуры на английский язык.

Далее терминал нам объясняет, чем он будет заниматься дальше.

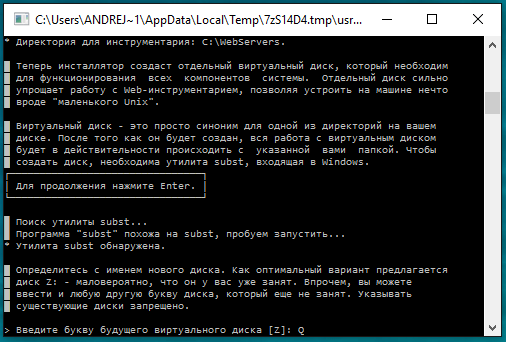
[](https://masalkin.name/upload/image_post/create/2016/04/22/big/6.png)Соглашаемся с установщиком денвера

Жмем Enter.

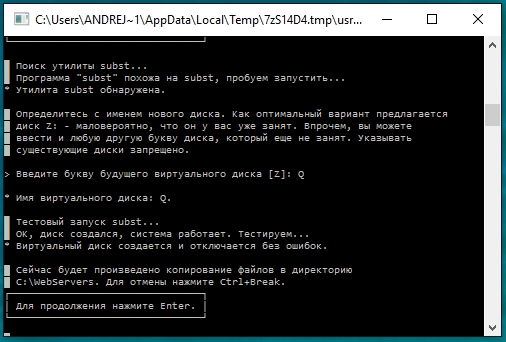
Нам будет предложено выбрать букву виртуального диска. Обычно предлагается последняя буква алфавита Z. Но если она у вас занята, то выбирайте какую-то другую. Так как Z у меня занята.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/7.png)Выбираю какая буква диска свободна

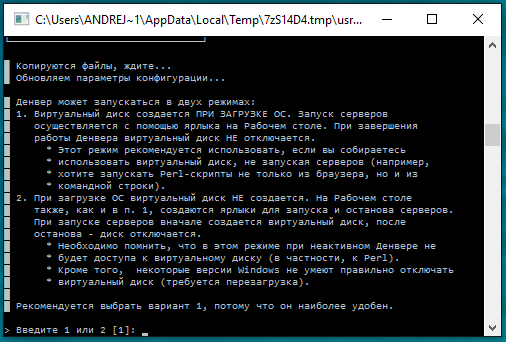
Я выбрал букву Q в качестве виртуального диска. Вписываю эту букву и жму Enter.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/8.png)Выбираю букву раздела для локального сервера

Программа проведет тестовый запуск и если все хорошо, то предложит нажать Enter, что вы и должны будете сделать.

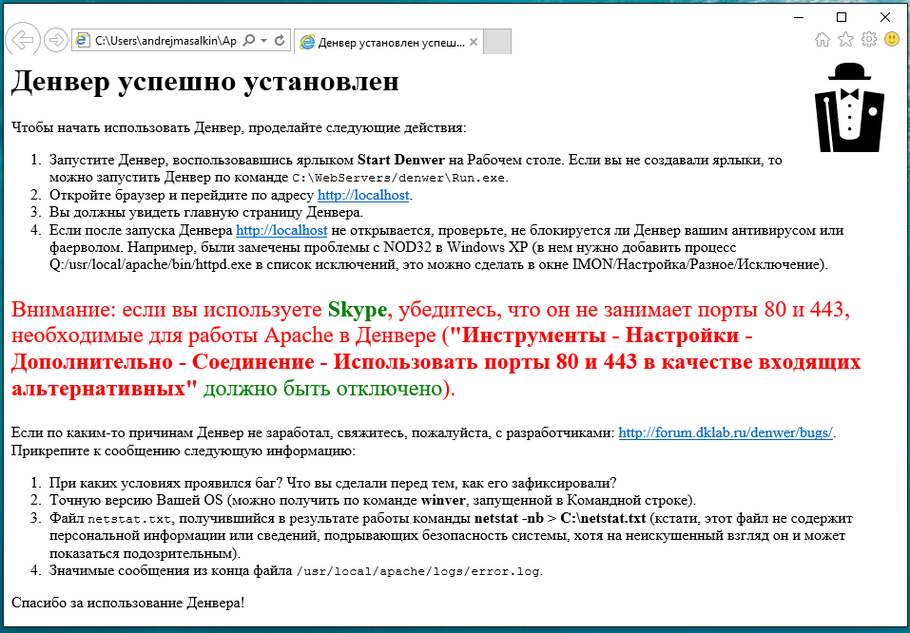
[](https://masalkin.name/upload/image_post/create/2016/04/22/big/9.png)

После этого программа начнет копировать все нужные файлы в соответствующую директорию. Когда копирование закончится, вам будет предложено выбрать 2 варианта запуска Денвера. И рекомендация использовать 1 вариант.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/10.png)

Какой вариант Денвера установить?

Использовать вариант под 1 номером. Указываете тот, что вы хотите, и жмете Enter.

Далее будет предложено установить ярлыки на рабочий стол. [](https://masalkin.name/upload/image_post/create/2016/04/22/big/11.png)Установка Денвера прошла успешно

И так, после установки у вас на рабочем столе должны появиться вот эти ярлыки.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/12.png)Новые ярлыки на рабочем столе

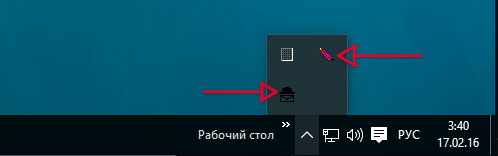
Обратите внимание на рекомендации, которые написаны в браузере после установки Денвера. Они предостерегут вас от возникновения проблем в работоспособности веб-сервера.

Запуск Денвера.

И так, чтобы начать работу с Денвером, его необходимо запустить. Для этого используйте ярлык Start Denwer. После первого запуска можете увидеть вот такую картину.

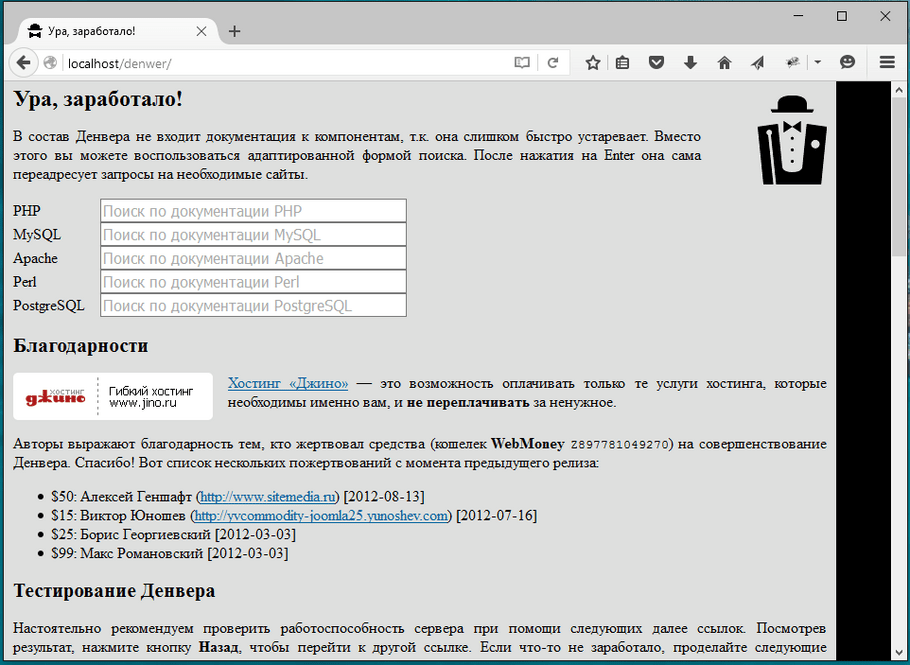
[](https://masalkin.name/upload/image_post/create/2016/04/22/big/13.png)Запускаем Denwer

Чтобы убедиться, что Denwer работает, загляните в трей. Там должна появиться вот такие иконки.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/14.png)

После чего открывайте браузер и в адресной строке набирайте http://localhost.

В результате должны увидеть эту страницу.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/15.png)

Это означает, что все работает прекрасно.

Настройка Denwer.

Для комфортной работы с сайтами на локальном диске, следует внести некоторые изменения в файлы конфигурации PHP. Для этого открываем файл php.ini, который можно найти по следующему пути C:\WebServersusr\local\php5

Этот файл открываем обычным блокнотом, и ищем следующие строки.

max*execution*time = 30, меняем значение на 180.

max*input*time = 60, меняем так же на 180.

memory*limit = 128MB, меняем на 512M.*

post*max*size = 8M, меняем на 999M

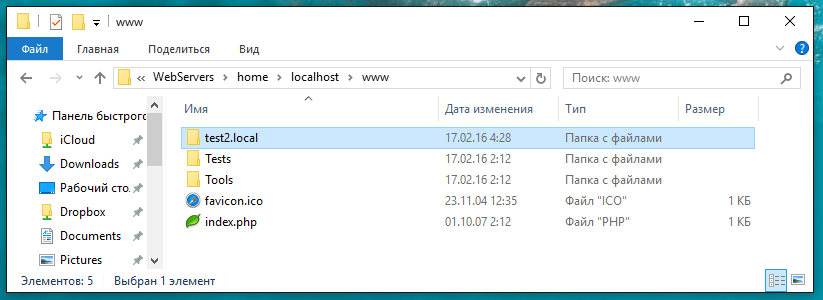
Теперь сохраняем все изменения и перезагружаем Денвер.

Где размещать сайты в Денвер?

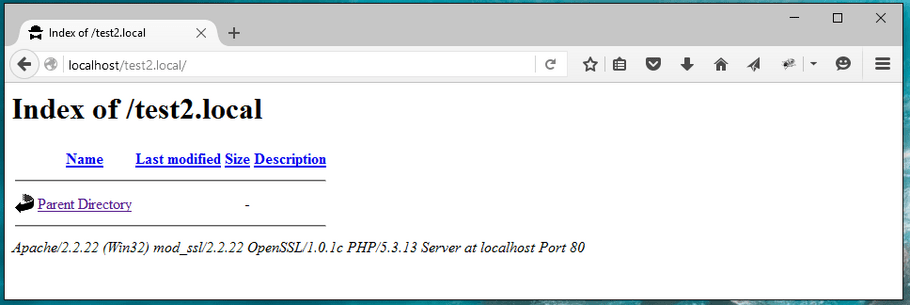
Все свои сайты вы будете размещать вот по этому пути C:\WebServers\home\localhost\www

То есть в папке www вы создаете папку test2.local и размещаете там либо свои HTML файлы, либо файлы движка. В следующем уроке мы поговори о том, как установить WordPress на Denwer, поэтому станет более понятным.

А сейчас в качестве примера я создал эту папку.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/19.png)

И если при обращении в адресной строке localhost/test2.local мы видим такую картину, то значит, все работает отлично.

[](https://masalkin.name/upload/image_post/create/2016/04/22/big/20.png)

Как удалить Denwer

Так как Денвер не затрагивает системных файлов, то его удаление сводится к простому переносу в корзину установочной папки. Но перед тем как удалять Денвер, остановите его работы. Отключите виртуальный диск. И удалите ярлыки с рабочего стола. После чего смело перетаскивайте папку WebServers, которую установили, в корзину.

Какие есть альтернативы Denwer’у

Если вдруг вы столкнетесь с трудностями в установке Денвера. Вдруг он не будет запускаться, либо еще какие проблемы. То ват для вас список подобных пакетов, которые вы можете установить и использовать в качестве виртуального сервера.

* XAMPP ([статья о том как его установить](https://masalkin.name/ustanovka-i-nastrojka-xampp-vmesto-denwer-a/))
* MAMP PRO (платный софт, есть для MAC OS)
* WAMP
* AppServ
* Open Server

**Настройка конфигураций Apache, PHP и MySQL**

*Цель работы*: получить практические навыки развертывания и базовой настройки АМР-платформы, изучить функции PHP для получения данных формы, работы со строками и массивами.

*Теоретические сведения*

Виртуальная платформа AMP = Apache + MySQL + PHP.

АМР это не совсем платформа, скорее среда выполнения. Но под неё написано большинство WEB-приложений. Поэтому решено, что AMP – это платформа для WEB-приложений. На AMP работают CMS, различные движки блогов, Интернет магазинов, такие как Joomla, WordPress. Все WEB приложения работают в определённой среде, необязательно AMP, это может быть и среда Tomcat, GlassFish, JBoss при использовании Java.

Apache является [кроссплатформенным ПО](http://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%BE%D1%81%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), поддерживая операционные системы [GNU/Linux](http://ru.wikipedia.org/wiki/GNU/Linux), [BSD](http://ru.wikipedia.org/wiki/BSD), [Mac OS](http://ru.wikipedia.org/wiki/Mac_OS), [Microsoft Windows](http://ru.wikipedia.org/wiki/Microsoft_Windows), [Novell NetWare](http://ru.wikipedia.org/wiki/Novell_NetWare), [BeOS](http://ru.wikipedia.org/wiki/BeOS).

Программа на PHP (да и на любом другом языке программирования) – это набор команд (инструкций). Обработчику программы (парсеру) необходимо как-то отличать одну команду от другой. Для этого используются специальные символы – разделители. В PHP инструкции разделяются так же, как и в Cи или Perl, – каждое выражение заканчивается точкой с запятой.

Закрывающий тег «?>» также подразумевает конец инструкции, поэтому перед ним точку с запятой не ставят. Например, два следующих фрагмента кода эквивалентны:

1.php (расположить на сервере Webservers/home/localhost/www/1.php)

<?php

echo "Hello, world!"; // точка с запятой в конце команды обязательна

?>

Переменная в PHP обозначается знаком доллара, за которым следует ее имя. Например:

$my\_var

Имя переменной чувствительно к регистру, т.е. переменные $my\_var и $My\_var различны.

Имена переменных соответствуют тем же правилам, что и остальные наименования в PHP: правильное имя переменной должно начинаться с буквы или символа подчеркивания с последующими в любом количестве буквами, цифрами или символами подчеркивания.

<?php

$first = ' Text '; // Присваиваем $first значение ' Text '

$second = $first; // Присваиваем $second значение переменной $first

$first = ' New text '; // Изменяем значение $first на ' New text '

echo "Переменная с именем first " .

"равна $first <br>"; // выводим значение $first

echo "Переменная с именем second " .

"равна $second"; // выводим значение $second

?>

*Результат работы этого скрипта будет следующим:*

Переменная с именем first равна New text

Переменная с именем second равна Text

Массив – это тип данных, с данными этого типа должны быть определены операции. Какие же операции можно производить с массивами? Массивы можно складывать и сравнивать.

Определить массив можно с помощью конструкции array() или непосредственно задавая значения его элементам.

array ([key] => value,

[key1] => value1, ... )

Языковая конструкция array() принимает в качестве параметров пары ключ => значение, разделенные запятыми. Символ => устанавливает соответствие между значением и его ключом. Ключ может быть как целым числом, так и строкой, а значение может быть любого имеющегося в PHP типа. Числовой ключ массива часто называют индексом. Индексирование массива в PHP начинается с нуля. Значение элемента массива можно получить, указав после имени массива в квадратных скобках ключ искомого элемента.

Складывают массивы с помощью стандартного оператора «+». Вообще говоря, эту операцию по отношению к массивам точнее назвать объединением. Если у нас есть два массива, $a и $b, то результатом их сложения (объединения) будет массив $c, состоящий из элементов $a, к которым справа дописаны элементы массива $b. Причем, если встречаются совпадающие ключи, то в результирующий массив включается элемент из первого массива, т.е. из $a. Таким образом, если складываются массивы в языке PHP, от перемены мест слагаемых сумма меняется.

<?

$a = array("и"=>"Информатика", "м"=>"Математика");

$b = array("и"=>"История","м"=>"Биология", "ф"=>"Физика");

$c = $a + $b;

$d = $b + $a;

print\_r($c);

/\* получим: Array([и]=>Информатика [м]=>Математика [ф]=>Физика) \*/

print\_r($d);

/\* получим: Array([и]=>История [м]=>Биология [ф]=>Физика) \*/

?>

Массив в PHP представляет собой упорядоченную карту – тип, который преобразует значения в ключи. Этот тип оптимизирован в нескольких направлениях, поэтому вы можете использовать его как собственно массив, список (вектор), хеш-таблицу (являющуюся реализацией карты), стэк, очередь и т.д. Поскольку вы можете иметь в качестве значения другой массив PHP, можно также легко эмулировать деревья.

*Функция in\_array*

in\_array("искомое значение","массив",

["ограничение на тип"]);

позволяет установить, содержится ли в заданном массиве искомое значение. Если третий аргумент задан как true, то в массиве нужно найти элемент, совпадающий с искомым не только по значению, но и по типу. Если искомое значение – строка, то сравнение чувствительно к регистру.

Например, имеется массив не изученных нами языков программирования. Мы хотим узнать, содержится ли в этом массиве язык PHP. Напишем следующую программу:

<?php

$langs = array("Lisp","Python","Java",

"PHP","Perl");

if (in\_array("PHP",$langs,true))

echo "Надо изучить PHP<br>"; // выведет сообщение "Надо изучить PHP"

if (in\_array("php",$langs))

echo "Надо бы изучить php<br>";

// ничего не выведет, поскольку в массиве есть строка "PHP", а не "php"

?>

В качестве искомого значения этой функции может выступать и массив. Правда, это свойство было добавлено только начиная с PHP 4.2.0.

Например:

<?php

$langs = array("Lisp","Python",array("PHP","Java"),"Perl");

if (in\_array(array("PHP","Java"),$langs))

echo "Надо бы изучить PHP и Java<br>";

?>

*Функция array\_search*

Это еще одна функция для поиска значения в массиве. В отличие от in\_array в результате работы array\_search возвращает значение ключа, если элемент найден, и ложь – в противном случае. А вот синтаксис у этих функций одинаковый:

array\_search("искомое значение","массив",

["ограничение на тип"]);

Сравнение строк чувствительно к регистру, а если указан опциональный аргумент, то сравниваются еще и типы значений. До PHP 4.2.0, если искомое значение не было найдено, эта функция возвращала ошибку или пустое значение NULL.

<?php

$langs = array("","Lisp","Python","Java", "PHP","Perl");

if (!array\_search("PHP",$langs))

echo "Надо бы изучить PHP<br>";

else {

$k = array\_search("PHP",$langs);

echo "PHP я изучил $k-м";

}

?>

В результате мы получим строчку*:* PHP я изучил 4-м

Очевидно, что эта функция более функциональна, чем in\_array, поскольку мы не только получаем информацию о том, что искомый элемент в массиве есть, но и узнаем, где именно в массиве он находится. А что будет, если искомых элементов в массиве несколько? В таком случае функция array\_search() вернет ключ первого из найденных элементов. Чтобы получить ключи всех элементов, нужно воспользоваться функцией array\_keys().

*Считывание данных формы:*

my1.php

<?php

print "hello! ";

?>

<form action=my1.php method=post>

Вввод имени <input type=text name=login>

<br> <input type=submit name=ok value=Отправить>

</form>

<?php

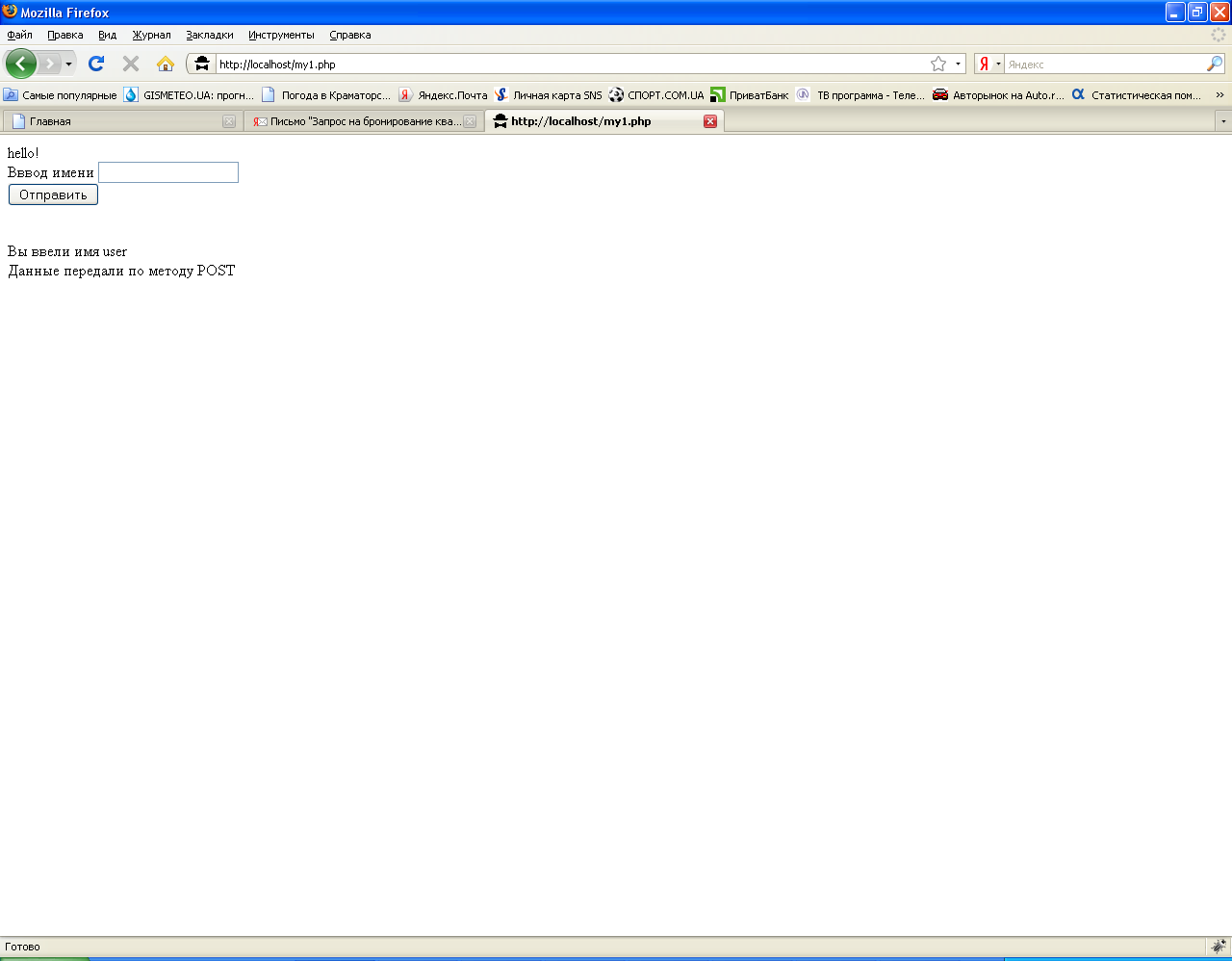
$method = $\_SERVER["REQUEST\_METHOD"]; // получить метод

$name=$\_REQUEST["login"]; // получение данных из поля login

echo "<br>Вы ввели имя $name " ;

echo "<br>Данные передали по методу $method";

?>



*Задание к работе:*

1. Ознакомиться с теоретическим материалом.
2. Cоздать скрипт на php для выполнения ввода информации в элементы формы, обработать введены данные, выдать информацию пользователю. Использовать элементы диалога таблицы 1.
3. Оформить отчет согласно требованиям.

*Таблица 1- Индивидуальное задание для выбора элементов диалога*

|  |  |  |  |
| --- | --- | --- | --- |
| *№* | *Текст задания* | *№* | *Текст задания* |
| 1 | Text, password, textarea | 7 | Radio, select |
| 2 | Select, textarea | 8 | Text, password |
| 3 | radio, textarea | 9 | reset, select |
| 4 | Image (кнопка), textarea | 10 | Text, radio |
| 5 | Text, select, reset | 11 | Select, text |
| 6 | Text, checkbox | 12 | Radio, text |

*Отчет должен содержать:*

1. Название и цель работы.
2. Ход работы с детальным описанием выполненных действий с рисунками, листингом кода.
3. Экранные формы браузера с загруженными страницами.
4. Выводы о проделанной работе.

*Вопросы для подготовки к защите работы:*

1. Какие элементы диалога можно располагать на форме?
2. Методы передачи данных формы?
3. Как обработать данные, введенные в элементы диалога?
4. Можно ли передавать данные формы разным модулям? Как организовать такой вид передачи?
5. Какие вы знаете функции для работы со строками?

**Вычисление значений выражений по формуле в среде программирования**

*Цель*: Изучение основ программирования на языке , разработка программ с использование линейных алгоритмов, и арифметических операций. Разработка, компиляция, отладка программ

*Задания для практической работы:*

1. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_5.gif

2. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_6.gif

3. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_7.gif

4. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_8.gif

5. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_9.gif

6. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_10.gif

7. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_11.gif

8. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_12.gif

9. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_13.gif

10. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_14.gif

11. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_15.gif

12. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_16.gif

13. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_17.gif

14. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_18.gif

15. Найдите значение выражения: http://csaa.ru/wp-content/uploads/2018/01/vychislenie-znachenij-matematicheskih-vyrazhenij-2_19.gif

.Дать определения следующий понятий, привести примеры:

* алгоритм;
* алгоритмический язык;
* язык программирования;
* синтаксическое правило;
* семантика;
* машинный код;
* язык высокого уровня;
* структурное программирование;
* компилятор;
* интерпретатор;
* алфавит языка;
* идентификатор;
* ключевое слово;
* числовая константа;
* комментарий;
* переменная;
* именованная константа;
* инициализатор;
* тип данных;
* интегрированная среда разработки;

**Организация вывода данных в PHP**

*Цель*: Научится выводить данные в браузер.

*Ход работы:*

Вывод данных осуществляется по принципу PHP -> Сервер ->Браузер. Существует две главных функции**print** и **echo**. Нельзя сказать, что это функции, так как функции это немного другое. Это конструкции языка. Но сделаем такое пренебрежение в разговоре.

1.<?php

2.print ("Вывод текста через print");

3.echo "Вывод текста через echo";

4.?>

В первом уроке мы создали файл echo.php . Откройте его и напишите следующий код:

1.<h1>

2.<?php

3.echo "Заголовок";

4.?>

5.</h1>

Выдастся белый экран с надписью “Заголовок”. Отсюда моно сделать вывод, что php код можно вставлять в html теги страницы. Данный код можно записать по-другому:

1.<?php

2.echo "<h1>Заголовок</h1>";

3.?>

**Первый и второй вариант одинаковые для вывода данных**. Можно укоротить запись таким образом:

1.<h1>

2.<?=" Заголовок ";?>

3.</h1>

Это будет абсолютно тоже самое. Рассмотрим внимательнее спец. cимволы **php echo** можно заменить на **=**

1.<h1>

2.<?php echo "Текст"?>

3.</h1>

4.//Тоже самое!

5.<h1>

6.<?="Текст"?>

7.</h1>

Данные можно записывать в переменные. Об этом поговорим в следующем уроке.

**ЗАДАНИЕ 1**

- Создайте переменную $name и присвойте ей значение содержащее Ваше имя, например "Владимир"(обязательно в кавычках!)

- Создайте переменную $age и присвойте ей значение содержащее Ваш возраст, например 25

**ЗАДАНИЕ 2**

- Выведите с помощью echo(или print) фразу "Меня зовут: ваше\_имя"

- Выведите фразу "Мне ваш\_возраст лет", например: "Мне 25 лет"

- Удалите переменную $age.

***Решение:***

01.

02.<?php

03.$name = "Владимир";

04.$age = 25;

05.?>

06.

07.<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

08.<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ru" lang="ru">

09.<head>

10.<title>Переменные и вывод</title>

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

12.</head>

13.<body>

14.

15.<h1>Переменные и вывод</h1>

16.<?php

17.echo "Меня зовут: $name", "<br />";

18.echo "Мне $age лет";

19.unset($age);

20.?>

21.

22.</body>

23.</html>

**Составление линейных программ на PHP**

*Цель: п*риобрести практические навыки в составлении, отладке линейных программ и в работе со средой программирования.

**Задание 1**. Даны два числа 5 и 7. Найти их сумму и произведение.

**<?php**  
$a = 5;  
$b = 7;  
$c = $a + $b;  
$d = $a\*$b;  
echo $c, $d;

**?>**

**Задание 2** Даны два числа 4 и 6. Найдите сумму их квадратов.

**<?php**  
$a = 4;  
$b = 6;  
$c = $a\*$a + $b\*$b;  
echo $c;

**?>**

**Задание 3.** Даны три числа 3, 5, 8. Найдите их среднее арифметическое.

**<?php**  
$a = 3;  
$b = 5;  
$c = 8;  
$d = ($a + $b + $c)/3  
echo $d;

**?>**

**Задание 4.** Даны три числа x = 2,y = 6 и z = 9. Найдите (x+1)4?2(z?2x^2+y^2)

**<?php**  
$x = 2;  
$y = 6;

$z = 9;  
$a = ($x+1)\*4 – 2 ($z – 2\*$x\*$x+$y\*$y);  
echo $a

**?>**

**Задание 5.** Даны три ненулевых числа $a = 4, $b = 8, $c = 3. Найдите всевозможные результаты деления суммы двух из них на оставшееся третье число.

**<?php**  
 $a = 4;  
$b = 8;  
$c = 3;  
$d = ($a +$b)/$c;  
$r = ($b+$c)/$a;  
$f = ($a + $c)/$b;  
echo $d,$r,$f;

**?>**

**Задание 6.** Дано два числа 17 и 54. Найдите сумму 40% от первого числа и 84% от  
    второго числа.

**<?php**  
$a = 17;  
$b = 54;  
$c = 0,4\*17 + 0,84\*54;  
echo $c;  
 **?>**

**Задание 7.** Дано трехзначное числа. Найдите сумму его цифр.

**<?php**  
$var = 578; *// Объявляем число*  
$var .= ‘ ; *// Приводим его к строковому типу*  
echo $a = $var[2] + $var[1] + $var[0]; *// Выводим переменную равную сумме цифр числа*

**?>**

**Задание 8.** Переделайте приведенный код так, чтобы в нем использовались операции **+=, -=, \*=, /=, ++, --**. Количество строк кода при этом не должно измениться. Код для переделки:

<?php

$var **=** 1;

$var **=** $var **+** 12;

$var **=** $var **-** 14;

$var **=** $var **\*** 5;

$var **=** $var **/** 7;

$var **=** $var **+** 1;

$var **=** $var **-** 1;

echo $var;

?>

**Задание 9.** Создайте переменную $text и присвойте ей значение 'Привет, Мир!'. Выведите значение этой переменной на экран.

**Задание 10.** Создайте переменные $text1='Привет, ' и $text2='Мир!'. С помощью этих переменных и операции сложения строк выведите на экран фразу 'Привет, Мир!'.

**Задание 11.**  Создайте переменную $name и присвойте ей ваше имя. Выведите на экран фразу 'Привет, %Имя%!'. Вместо %Имя% должно стоять ваше имя.

**Задание 12.**  Создайте переменную $age и присвойте ей ваш возраст. Выведите на экран 'Мне %Возраст% лет!'.

**Задание 13.** Создайте переменную $text и присвойте ей значение 'abcde'. Обращаясь к отдельным символам этой строки выведите на экран символ 'a', символ 'c', символ 'e'.

**Задание 14.** Дана произвольная строка, например, 'abcde'. Поменяйте первую букву (то есть букву 'a') этой строки на '!'.

**Задание 15.** Создайте переменную $num и присвойте ей значение '12345'. Найдите сумму цифр этого числа.

**Задание 16.** Переделайте этот код так, чтобы в нем использовались операции ++ и --. Количество строк кода при этом не должно измениться.

$var = 10;

$var = $var + 1;

$var = $var + 1;

$var = $var - 1;

echo $var;

**Задание 17.** Переделайте этот код так, чтобы в нем использовались операции ++, -- , +=, -=, \*=, /=. Количество строк кода при этом не должно измениться.

$var = 10;

$var = $var + 7;

$var = $var + 1;

$var = $var - 1;

$var = $var + 12;

$var = $var \* 7;

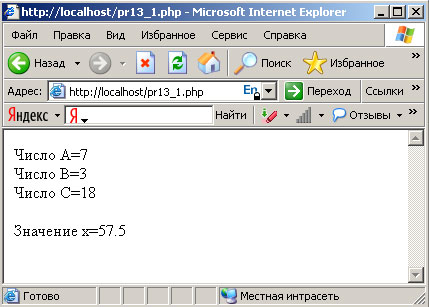
$var = $var - 15;

echo $var;

**Задание 18.** Создадим скрипт для решения следующей задачи: написать программу, которая для заданных значений **A, B, C** будет выводить результат выражения **x=A\*(B/6)+3\*C**.

Решением этой задачи будет следующий **php**-скрипт:

<?php

$A=7;

$B=3;

$C=18;

*// выводит значение переменной $A*

echo "Число A=".$A."<br>";

*// выводит значение переменной $B*

echo "Число B=".$B."<br>";

*// выводит значение переменной $C*

echo "Число C=".$C."<br><br>";

$x=$A\*($B/6)+3\*$C; *// вычисляет значение выражения*

echo "Значение x=".$x; *// выводит результат вычисления*

?>

**Задание 19.** Задавать значения A, B, C в самом php-скрипте непрактично, так как необходимо будет каждый раз открывать файл с программой, если нужно будет проверить ее работу при других значениях. Чтобы этого избежать изменим содержимое файла следующим образом: создадим текстовые поля для каждого значения. В результате получим следующую программу:

<html>

<head>

<title>Решение задачи</title>

<meta charset="utf-8">

</head>

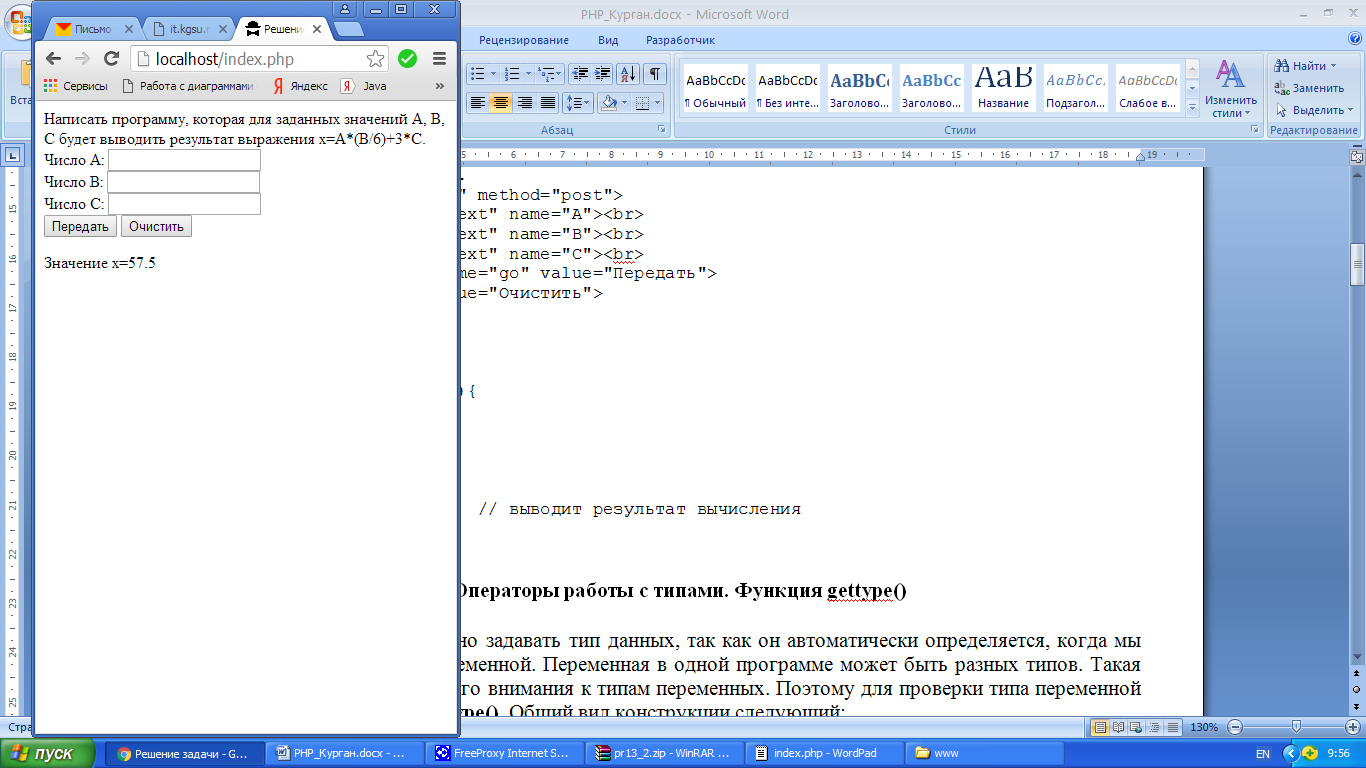
<body>

Написать программу, которая для заданных значений A, B, C будет выводить результат выражения x=A\*(B/6)+3\*C.

<form action="index.php" method="post">

Число A: <input type="text" name="A"><br>

Число B: <input type="text" name="B"><br>

Число C: <input type="text" name="C"><br>

<input type="submit" name="go" value="Передать">

<input type="reset" value="Очистить">

</form >

<?php

if (isset($\_POST['go'])){

$A=$\_POST['A'];

$B=$\_POST['B'];

$C=$\_POST['C'];

$x=$A\*($B/6)+3\*$C; }

echo "Значение x=".$x; // выводит результат вычисления

?>

</body></html>

**Задание 20. Решите задачи на простейшую арифметику, через ввод чисел:**

1. Даны два числа. Найти их сумму и произведение.
2. Даны два числа. Найдите сумму их квадратов.
3. Даны три числа. Найдите их среднее арифметическое.
4. Даны три числа x,y и z. Найдите (x+1)4−2(z−2x2+y2)+|siny|
5. Даны три ненулевых числа. Найдите всевозможные результаты деления суммы двух из них на оставшееся третье число.
6. Дано натуральное числа. Найдите остатки от деления этих чисел на 3 и на 5.
7. Дано число. Увеличьте его на 30%, на 120%.
8. Дано два числа. Найдите сумму 40% от первого числа и 84% от второго числа.
9. Дано трехзначное числа. Найдите сумму его цифр.
10. Дано трехзначное числа. Поменяйте среднюю цифру на ноль.
11. Найдите число, полученное выписыванием в обратном порядке цифр данного трехзначного натурального числа.

*Контрольные вопросы*

1. Порядок работы со средой программирования.

2. Организация ввода текста программы в память ЭВМ и его редактирование.

3. Порядок пуска и останова программы.

4. Формат оператора присваивания.

**Условный оператор (полная, неполная форма). Вложенное ветвление**

**Цель:** сформировать понятие о ветвлении в полной и неполной форме, начать формировать навык решения задач на использование в них разных форм ветвлений.

**Задание 1.** Дано число 15. Если оно больше 10, то увеличьте его на 100, иначе уменьшите   на 30.

**<?php**  
$a = 15;  
if ($a>10)

{$a = $a + 100};  
else  
{$a = $a – 30};  
echo $a **?>**

**Задание 2.** Дано натуральное число 8. Если оно четное, то уменьшите его в 2 раза, иначе увеличьте в 3 раза.

**<?php**

if ($number / 2 == ceil($number / 2))  
{$number = $number / 2;}

else { $number = $number \*3; }  
   echo $number;**?>**

**Задание 3.** Дано число. Если оно не меньше 50, то выведите квадрат этого числа, если же это число больше 10 и меньше 30, то выведите ноль, в остальных случаях выведите слово «Ошибка»

**<?php**  
$a = ee; *// Задаем число $a, например, 45*

if ($a > 50) { 4. echo $b=pow($a,2); }  
elseif ($a > 10 & $a < 30) { echo 0; }  
else { echo «Ошибка»; }**?>**

**Задание 4.** Дано два числа $a = 15, $b = 4. Вывести наибольшее из них.

**<?php**

$a = 15;

$b = 4;

if ($a > $b){ echo $a};

else { echo $b }; **?>**

**Задание 5.** Дано два числа $a = 19, $b = 143. Вывести ‘Да’, если они отличаются на 100,  иначе вывести ‘Нет’

**<?php**  
$a = 19;  
 $b = 143;

if ($b – $a = 100 ) { echo «No» }; else { echo «Yes» };**?>**

**Задание 6.** Дано два натуральных числа. Вывести ‘Да’, если они отличаются не более чем на 20, иначе вывести ‘Нет’.

**<?php**

$a = 15;  
$b = 30;  
if ($b – $a > 20){ echo «No» };  
else{echo «Yes»};**?>**

**Задание 7.** В переменной **$min** лежит число от 0 до 59. Определите в какую **четверть часа** попадает это число (в первую, вторую, третью или четвертую).

<?php

……

if ($min **>=** 0 and $min **<=** 14) {

echo 'В первую четверть.';}……?>

**Задание 8.** Если переменная $a равна или меньше 1, а переменная $b больше или равна 3, то выведите сумму этих переменных, иначе выведите их разность (результат вычитания). Проверьте работу скрипта при$a и $b, равном 1 и 3, 0 и 6, 3 и 5.

**Задание 9.** Если переменная $a больше нуля и меньше 5-ти, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при $a, равном 5, 0, -3, 2.

**Задание 10.** Если переменная $a равна нулю или равна двум, то прибавьте к ней 7, иначе поделите ее на 10. Выведите новое значение переменной на экран. Проверьте работу скрипта при $a, равном 5, 0, -3, 2.

**Задание 11.** Если переменная $a равна или меньше 1, а переменная $b больше или равна 3, то выведите сумму этих переменных, иначе выведите их разность (результат вычитания). Проверьте работу скрипта при $a и $b, равном 1 и 3, 0 и 6, 3 и 5.

**Задание 12.** Если переменная $a больше 2-х и меньше 11-ти, или переменная $b больше или равна 6-ти и меньше 14-ти, то выведите 'Верно', в противном случае выведите 'Неверно'

**Задание 14.** В переменной $day лежит какое-то число из интервала от 1 до 31. Определите в какую декаду месяца попадает это число (в первую, вторую или третью).

**Задание 15.** В переменной $month лежит какое-то число из интервала от 1 до 12. Определите в какую пору года попадает этот месяц (зима, лето, весна, осень).

**Задание 16.** В переменной $year хранится год. Определите, является ли он високосным (в таком году есть 29 февраля). Год будет високосным в двух случаях: либо он делится на 4, но при этом не делится на 100, либо делится на 400. Так, годы 1700, 1800 и 1900 не являются високосными, так как они делятся на 100 и не делятся на 400. Годы 1600 и 2000 - високосные, так как они делятся на 400.

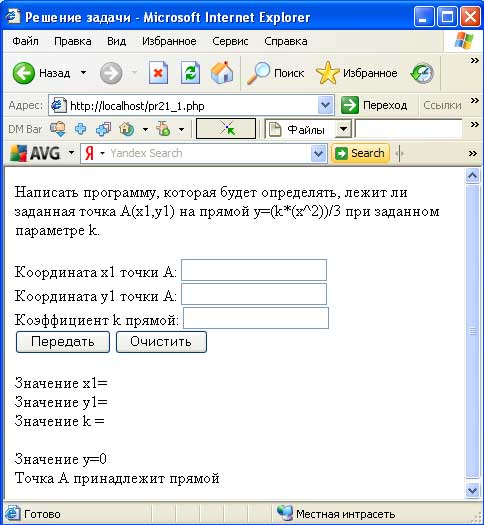
**Задание 17**. Решите задачи с условным оператором, данные вводятся с клавиатуры:

1. Дано число. Если оно больше 10, то увеличьте его на 100, иначе уменьшите на 30.
2. Дано натуральное число. Если оно четное, то уменьшите его в 2 раза, иначе увеличьте в 3 раза.
3. Дано число. Если оно не меньше 50, то выведите квадрат этого числа, если же это число больше 10 и меньше 30, то выведите ноль, в остальных случаях выведите слово "Ошибка"
4. Дано два числа. Вывести наибольшее из них.
5. Дано два числа. Вывести 'Да', если они отличаются на 100, иначе вывести 'Нет'
6. Дано два числа. Вывести 'Да', если они отличаются не более чем на 20, иначе вывести 'Нет'.
7. Дан номер месяца. Вывести название поры года (весна, лето и так далее) или слово 'Ошибка', если месяца с таким номером не существует.
8. Вычислите значение выражения x2−4y−1−−−−√sin2x+|x|. Перед вычислением проверить корректность значений переменных.
9. В данном трехзначном числе переставьте цифры так, чтобы новое число оказалось наибольшим из возможных.
10. Решите  уравнение ax2+bx+c=0.
11. Вычислите значение выражения max2(x,y,z)−3x⋅min(x,y,z)

sin1+max(x,y−z)/min(y,z).

Если вычисление невозможно, то сообщите об этом пользователю.

**Задание 18.** Приведем решение следующей задачи: написать программу, которая будет определять, лежит ли заданная точкаA(x1,y1) на прямой y=kx2/3 при заданном параметре k.

Решением этой задачи будет следующая программа:

<html>

<head>

<title>Решение задачи</title>

</head>

<body>

Написать программу, которая будет определять, лежит ли заданная точка A(x1,y1) на

прямой y=(k\*(x^2))/3 при заданном параметре k.

<form>

Координата x1 точки A: <input type="text" name="x1"><br>

Координата y1 точки A: <input type="text" name="y1"><br>

Коэффициент k прямой: <input type="text" name="k"><br>

<input type="submit" name="go" value="Передать">

<input type="reset" value="Очистить">

</form>

<?php

echo "Значение x1=".$x1."<br>";

echo "Значение y1=".$y1."<br>";

echo "Значение k =".$k."<br><br>";

$y=($k\*($x1\*$x1))/3;

echo "Значение y=".$y."<br>";

if ($y==$y1)

{ echo "Точка A принадлежит прямой"; }

else { echo "Точка A не принадлежит прямой"; }

?>

</body>

</html>

*Контрольные вопросы:*

1. Какой оператор используется для записи ветвления?
2. Что такое составной оператор?
3. Какой синтаксис имеет составной оператор?
4. Как в программе записывается условный оператор?

**Решение текстовых задач на ветвление**

*Цель:* овладение практическими навыками разработки и программирования вычислительного процесса, связанного с проверкой нескольких условий, использование в программах средств для проверки корректности ввода данных, получение дальнейших навыков по отладке и тестированию программ, выработка при решении поставленных задач таких профессионально значимых качеств, как самостоятельность, ответственность, точность, творческая инициатива.

1. Дана переменная а со строковым значением. Если а равна имени, то выводить *«Привет, имя!»*, если а равна пустому значению, то выводить *«Привет, незнакомец!»*
2. Выводить перевод цветов с английского языка на русский, проверяя значение переменной (в которой присвоен цвет: $a='blue')
3. В заданном тексте везде заменить слово мир на слово римлянин.
4. Напечатать самое длинное слово из текста.

**Решение задач с использованием оператора Выбора**

*Цель:* овладение практическими навыками разработки и программирования вычислительного процесса, связанного с оператором Выбора.

Часто вместо нескольких расположенных подряд инструкций if-else целесообразно воспользоваться специальной конструкцией выбора switch-case. Раcсмотрим пример на elseif.

01.<?php

02.$d = 2;

03.if ($d == 1)

04.print(" 1 ");

05.elseif ($d == 2)

06.print(" 2 ");

07.elseif ($d == 3)

08.print(" 3 ");

09.elseif ($d == 4)

10.print(" 4 ");

11.?>

Данный пример выведет значение при совпадение условия, но перебор всех сравнений будет идти пока не закончиться конструкция.  Логичнее было бы закончить выполнение кода, сразу как условие выполниться.  Конструкция switch-case как раз такой логикой и обладает.

Рассмотрим синтаксис конструкции switch-case:

01.<?php

02.switch (Переменная/Выражение){

03.case Значение 1:

04.Операторы 1;

05.[break;]

06.case Значение 2:

07.Операторы 2;

08.[break;]

09.[default: Оператор]

10.}

11.?>

Наш пример будет в конструкции switch-case выглядеть следующем образом:

01.<?php

02.$d = 2;

03.switch ($d) {

04.case 1:

05.print(" 1 ");

06.break;

07.case 2:

08.print(" 2 ");

09.break;

10.case 3:

11.print(" 3 ");

12.break;

13.case 4:

14.print(" 4 ");

15.break;

16.}

17.?>

break; - означает выйти из конструкции. То есть если переменная совпала, то сразу закончить выполнение конструкции.

Очень интересно ведет себя конструкция, если убрать break;

01.<?php

02.$d = 2;

03.switch ($d) {

04.case 1:

05.print(" 1 ");

06.case 2:

07.print(" 2 ");

08.case 3:

09.print(" 3 ");

10.case 4:

11.print(" 4 ");

12.}

13.?>

Конструкция начинает работать по принципу:

Увидел совпадение в case, и выполняю всё что есть, не смотря наследующие case!!! В итоге выдаст на экран -  2 3 4

Если Вы посмотрите еще раз вверх на пример синтаксиса конструкции. То увидите - default: Оператор. Данные строчки выполняются при условии, что не одно выражение не совпало с case.

Пример:

01.<?php

02.$d = 100;

03.switch ($d) {

04.case 1:

05.print(" 1 ");

06.break;

07.case 2:

08.print(" 2 ");

09.break;

10.case 3:

11.print(" 3 ");

12.break;

13.case 4:

14.print(" 4 ");

15.break;

16.default: print(" Переменная не совпала с case. Вывелось значение по умолчанию. ");

17.}

18.?>

Так же как у конструкции “if и elseif” есть альтернативный синтаксис, который часто используется. Вот пример его:

01.<?php

02.$x=3;

03.switch ($x):

04.case 0:

05.echo "x=0";

06.break;

07.case 1:

08.echo "x=1";

09.break;

10.case 2:

11.echo "x=2";

12.break;

13.default:

14.echo "x не равен 0, 1 или 2";

15.endswitch;

16.?>

В нем нет  ”{ }” для определение начала, а используется  “ :”

А для конца - endswitch;

1. Переменная $num может принимать одно из значений: 1, 2, 3 или 4. Если она имеет значение '1', то в переменную $result запишем 'зима', если имеет значение '2' – 'лето' и так далее. Решите задачу через switch-case
2. Составьте программу для определения числа дней в месяце, если даны: номер месяца N –целое число от 1 до 12, целое число А, равное 1 для високосного года и 0 в противном случае.
3. Составить программу, которая по номеру дня в месяце печатает день недели. Считаем, что 1–е число месяца –понедельник.
4. Составить программу, которая по номеру семестра печатает курс, к которому относится введенный семестр (1 и 2 семестр –1 курс, 3 и 4 семестр –2 курс и т. д.).
5. Дано целое число n, соответствующее количеству углов геометрической фигуры. Составить программу, которая по введенному числу n печатает название фигуры (например, при n=3 программа
6. напечатает «треугольник», при n=5 –«пятиугольник», при n>8 –«многоугольник»). В случае если вводится число меньше 2, выводится сообщение об ошибке.
7. Составить программу, которая печатает номера дней в месяце, если вводится день недели. Считаем, что 1–е число месяца –понедельник, в месяце 31 день. Выводить на экран словесное описание дня недели и соответствующие числа месяца (например, вводится число 2, на экране появляется: «Вторник –2, 9, 16, 23, 30»).
8. Напишите программу, которая вычисляет стоимость междугородного телефонного разговора, в зависимости от кода города: Владивосток (423) –стоимость 2,22 руб.; Москва (095) –1,00 руб.; урманск (815) –1,2 руб.; Самара (846) –1,3 руб. Вводится код города и количество минут. На экран выводится название города, цена минуты разговора и итоговая стоимость.

**Тестирование программ**

*Цель:* овладение практическими навыками программирования вычислительного процесса, связанного с проверкой нескольких условий, использование в программах средств для проверки корректности ввода данных, получение дальнейших навыков по отладке и тестированию программ.

1. *На вход программе поступают три целых числа. Выведите наибольшее из них (программа должна вывести ровно одно число).*
2. Пользователь вводит четыре числа, найдите из них максимальное.
3. С клавиатуры вводятся координаты точки x, y. Напишите программу, которая сообщает положение точки на координатной плоскости: «1 четверть», «2 четверть» и т.д.
4. Человек вводит в компьютер число. Если оно находится в интервале от 28 до 30, то нужно напечатать текст ПОПАЛ, если оно больше или равно 30 - то ПЕРЕЛЕТ, если оно находится на отрезке от 0 до 28, то НЕДОЛЕТ, если число меньше нуля — НЕ БЕЙ ПО СВОИМ
5. Мужчина заполняет в военкомате анкету и программа должна в зависимости от указанного им возраста выводить разные подсказки, а именно:
6. Если указан возраст от 18 и до 27 лет, то сообщать, что заполняющий подлежит призыву на срочную службу или может служить по контракту.
7. Если указан возраст от 28 до 59 лет, то сообщать, что заполняющий может служить по контракту.
8. Если указан возраст менее 18 или более 59 лет, то сообщать о том, что заполняющий находится в непризывном возрасте.
9. Если указан неположительный возраст или возраст более 100 лет, то сообщить об ошибке.
10. Получите от пользователя два целых числа, если первое больше второго, то выведите на экран их сумму, а затем их разность, иначе:
11. сначала сравните их сумму с числом 2525, если она больше, то выведите на экран первое число, а если меньше, то второе.
12. затем выведите на экран второе число в квадрате
13. Коля набрал kk баллов, Миша nn баллов, а Петя dd баллов в ходе соревнований. Если:
14. сумма баллов Коли и Миши больше 2020
15. и сумма баллов Пети и Миши меньше 5555

то тогда вывести в консоль сообщение о том, что ничья, в противном случае вывести сообщение, что победил Коля.

1. Создайте переменную $day и присвойте ей произвольное числовое значение. С помощью конструкции switch выведите фразу *«Это рабочий день»*, если значение переменной $day попадает в диапазон чисел от 1 до 5 (включительно). Выведите фразу *«Это выходной день»*, если значение переменной $day равно числам 6 или 7. Выведите фразу *«Неизвестный день»*, если значение переменной $day не попадает в диапазон чисел от 1 до 7(включительно)

**Решение задач на циклы.**

*Цель –* формирование навыков практического применения приемов и способов решения задач циклических алгоритмов.

1. Выведите столбец чисел от 1 до 100.

|  |  |
| --- | --- |
| <?php  $i **=** 1;  while ($i **<=** 100)  { echo $i.'<br>'; $i**++**; }  ?> | <?php  for ($i **=** 1; $i **<=** 100; $i**++**)  { echo $i.'<br>'; }  ?> |

1. Выведите столбец чисел от 11 до 33
2. Выведите столбец четных чисел в промежутке от 0 до 100.
3. С помощью цикла найдите сумму чисел от 1 до 100

|  |  |
| --- | --- |
| <?php  $sum **=** 0; $i **=** 1;  while ($i **<=** 100)  { $sum **+=** $i; $i**++**; }  ?> | <?php  $sum **=** 0;  for ($i **=** 1; $i **<=** 100; $i**++**)  { $sum **+=** $i; }  ?> |

1. С помощью цикла for сформируйте строку '123456789' и запишите ее в переменную $str.

<?php

$str = '';

for ($i = 1; $i <= 9; $i++)

{ $str .= $i; }

echo $str; ?>

1. Числа Фибоначи - это ряд чисел, в котором каждое последующее число равно сумме двух предыдущих чисел.

Выведите на экран первые 100 чисел.

<?  
$a = 0;  
$b = 1;  
echo $a."<br>".$b."<br>";  
for ($i=0; $i<=98; $i++) {  
$c = $a+$b;  
echo $c."<br>";  
$a = $b;  
$b = $c;}  
?>

1. Вывести на экран все счастливые билеты. Счастливый билет - это когда сумма первых трех цифр в номере билета равна сумме второй тройки цифр (номер шестизначный).

<?php  
while($i < 1000000){  
/\* Обнуляем переменную result и находим количество нулей, которые нужно дописать спереди переменной i \*/  
$result = "";  
$zero = 6 - strlen($i);  
/\* Пишем к-ство нулей в переменную result. К примеру i = 22, то в result запишет 4 нуля, исходя из формулы выше \*/  
for($zero; $zero > 0; $zero--){$result .= 0;}  
/\* Дописываем вконце переменной result переменную i. К примеру i = 22, то допишем к result число 22 и будет 000022 \*/  
$result .= $i;  
/\* Сумируем первые три числа и последний три числа в переменные firstHalf и seconHalf \*/  
$firstHalf = $result[0] + $result[1] + $result[2];  
$secondHalf = $result[3] + $result[4] + $result[5];  
/\* Сравниваем значения в переменных выше, если ровно - это счастливый билет \*/  
if($firstHalf == $secondHalf){echo $result." - is a happy ticket! <br>";}  
/\* Инкрементируем счетчик, чтобы не получить зацикленную функцию \*/  
$i++;}  
?>

1. Напишите программу, определяющую сумму всех нечетных чисел от 1 до 99
2. Вывести числа в следущей последовательности: 200 199 198 … 0. Задание решите с помощью цикла do…while
3. Вывести числа в следущей последовательности: 200 199 198 … 0. Задание решите с помощью цикла do…while
4. Вывод квадратов натуральных чисел
5. Выведите на экран 10 случайных чисел от 1 до 100

<?php

echo mt\_rand(1, 100);

?>

1. Даны натуральные числа от 20 до 50. Напечатать те из них, которые делятся на 3, но не делятся на 5.
2. Даны натуральные числа от 35 до 87. Найти и напечатать те из них, которые при делении на 7 дают остаток 1, 2 или 5.
3. Даны натуральные числа от 1 до 50. Найти сумму тех из них, которые делятся на 5 или на 7.
4. Напечатать те из двузначных чисел, которые делятся на 4, но не делятся на 6.
5. Найти произведение двузначных нечетных чисел, кратных 13.
6. Создайте цикл, который выводит числа то 0 до 100 в HTML-элементах div; окраска HTML-элементов должна чередоваться («зебра»).
7. Выведите  10 раз фразу "You are welcome!"
8. Если переменная $a равна или меньше 1, а переменная $b больше или равна 3, то выведите сумму этих переменных, иначе выведите их разность (результат вычитания). Проверьте работу скрипта при$a и $b, равном 1 и 3, 0 и 6, 3 и 5.
9. Нарисуйте пирамиду, как показано на рисунке, только у вашей пирамиды должно быть 20 рядов, а не 5:

x  
xx  
xxx  
xxxx  
xxxxx

1. Нарисуйте пирамиду, как показано на рисунке, воспользовавшись циклом for

xx  
xxxx  
xxxxxx  
xxxxxxxx  
xxxxxxxxxx

*Контрольные вопросы:*

1. Что такое циклический алгоритм?
2. Какие обозначения используются в графическом представлении алгоритма для обозначения цикла?
3. Какие бывают циклы?
4. Что такое и когда используется цикл с предусловием?
5. Какой оператор в языке Паскаль используется для представления цикла с предусловием? Как в блок-схемах изображаются цикл с предусловием?
6. Особенности использования цикла с предусловием.
7. Что такое и когда используется цикл с постусловием?

**Тестирование программ. Составление отчетной документации**

*Цель –* формирование навыков практического применения тестирования программ по решению задач циклического алгоритма и составления отчетной документации (трассировочной таблицы)

Задание: протестировать задачи и составить трассировочные таблицы.

1. Выведите на экран n раз фразу "Silence is golden". Число n вводит пользователь на форме. Если n некорректно, вывести фразу "Bad n".
2. Найти сумму  1+4+7+10+...+112. Ответ: 2147
3. Решите числовой ребус ЛИСА + ВОЛК = ЗВЕРИ. Каждая буква соответствует своей цифре, причем различным буквам соответствуют различные цифры.
4. Найти сумму натуральных чисел от a до b, где a и b вводит пользователь. В случае некорректных a и b (например, a>b) вывести сообщение 'Сумма не существует'.
5. Вывести все числа, меньшие 10000, у которых есть хотя бы одна цифра 3 и которые не делятся на 5.
6. Вывести на экран квадрат, состоящий из n x n квадратов различного цвета.
7. Вывести 3 случайных числа от 0 до 100 без повторений.
8. Вывести на экран все шестизначные счастливые билеты. Билет называется счастливым, если сумма первых трех цифр в номере билета равна сумме последних трех цифр. Найдите количество счастливых билетов и  процент от общего числа билетов.
9. Найдите количество прямоугольных треугольников со сторонами, меньшими 100.
10. Вывести на черном фоне n красных квадратов случайного размера в случайной позиции в браузере.
11. Вывести n случайных серий из чисел 0, 1 и 2 таких, что сумма чисел в каждой серии равна s. Найти количество единиц в каждой серии, количество двоек в каждой серии, длину каждой серии, среднее количество двоек в сериях, среднюю длину серий и наибольшее количество ненулевых чисел в сериях. В случае некорректности n и k вывести слово 'Ошибка'.
12. В переменной $min лежит число от 0 до 59. Определите в какую четверть часа попадает это число (в первую, вторую, третью или четвертую).

<?php

……

if ($min >= 0 and $min <= 14) {

echo 'В первую четверть.';}

……

?>

*Контрольные вопросы:*

1. Что такое циклический алгоритм?
2. Какие бывают циклы?
3. Что такое трассировочная таблица?
4. Какие часто встречаемые ошибки при отладки программ?

**Решение задач с использованием циклов foreach. Тестирование и оптимизация работы программ**

*Цель*: Освоение приемов организации циклов foreach в программе на алгоритмическом языке, тестирование и оптимизация программ с помощью циклов.

**Задача 1.** Дан массив с элементами **'html', 'css', 'php', 'js', 'jq'**. С помощью цикла **foreach** выведите эти слова в столбик.

**Решение:** Для начала необходимо создать сам массив. В данном случае мы можем воспользоваться различными способами: объявить через **[]** либо просто воспользоваться присваиванием **$arr[] = 'html'; $arr[] = 'php';** и так далее. Я выбрал **первый способ**, так как он занимает меньше места при записи:

<?php

$arr **=** ['html', 'css', 'php', 'js', 'jq'];

?>

Теперь необходимо воспользоваться циклом **foreach**:

<?php

$arr **=** ['html', 'css', 'php', 'js', 'jq'];

foreach ($arr as $elem) {

echo $elem.'<br>';

}

?>

Как это все работает: в переменной **$elem** все элементы будут лежать по порядку: сначала **'html'**, потом **'css'** и так далее. Я буду выводить их на экран с помощью **echo**, а после вывода элемента ставить тег **<br>** (он делает перевод строки). Содержимое **$elem** и тег **<br>** являются строками, поэтому между ними я **ставлю точку** - соединяю тем самым две строки вместе.

**Задача 2.** Дан массив с элементами **10, 20, 15, 17, 24, 35**. Найдите сумму элементов этого массива. Запишите ее в переменную **$result**.

**Решение:** Первое решение, которое может прийти в голову новичку - это просуммировать все элементы массива 'руками': **$arr[0]+$arr[1]+...+$arr[5]**.

Однако, этот способ не правильный, так как в случае изменения массива **код придется переписывать** (например, в него добавится еще 10 элементов, или 100, что вообще будет печально). Решение необходимо делать как можно более универсальным.

Поэтому мы должны воспользоваться циклом **foreach** - в этом случае мы не будем привязаны к количеству элементов массива.

Теперь нужно подумать о том, как просуммировать элементы массива в цикле. Для этого я введу переменную **$result**, к которой при каждом проходе цикла буду прибавлять ее саму и еще один элемент массива таким образом **$result = $result + $elem**:

<?php

$arr **=** [10, 20, 15, 17, 24, 35];

$result **=** 0;

foreach ($arr as $elem) {

$result **=** $result **+** $elem;

}

echo $result; //выведем сумму на экран

?>

В строке **$result = 0** я зануляю переменную **$result**, если этого не сделать - в цикле возникнет ошибка в строке **$result = $result + $elem**, так как при первом проходе цикла переменная **$result** не будет определена.

Как работает строчка **$result = $result + $elem**: при первом проходе в переменной **$result** будет 0, в **$elem** – 10 (первый элемент массива), тогда в **$result** запишется 0+10 = 10.

При втором проходе цикла в **$result** лежит уже 10 (значение из прошлого прохода цикла), а в **$elem**лежит 20 (следующий элемент массива), тогда в **$result** запишется 10+20=30.

При третьем проходе цикла в **$result** лежит 30, а в **$elem** – 15, в **$result** запишется 30+15 = 45. И так далее пока цикл не закончится.

Строчку **$result = $result + $elem** можно записать короче: **$result += $elem**.

**Задача 3**. Дан массив с элементами **'html', 'css', 'php', 'js', 'jq'**. С помощью цикла **foreach** выведите эти слова в столбик.

**Задача 4.** Дан массив с элементами **1, 2, 3, 4, 5**. С помощью цикла **foreach** найдите **сумму** элементов этого массива. Запишите ее в переменную **$result**.

**Задача 5.** Дан массив с элементами **1, 2, 3, 4, 5**. С помощью цикла **foreach** найдите **сумму квадратов**элементов этого массива. Результат запишите переменную **$result**.

**Задача 6.** Дан массив **$arr**. С помощью цикла **foreach** выведите на экран столбец ключей и элементов в формате 'green - зеленый'.

$arr **=** ['green'**=>**'зеленый', 'red'**=>**'красный','blue'**=>**'голубой'];

**Задача 7.** Дан массив **$arr** с ключами **'Коля', 'Вася', 'Петя'** и с элементами **'200', '300', '400'**. С помощью цикла **foreach** выведите на экран столбец строк такого формата: *'Коля - зарплата 200 долларов.'*.

*Контрольные вопросы:*

1. Что такое циклический алгоритм foreach?
2. Что такое трассировочная таблица?
3. Какие ошибки встретились при отладки программ?

**Применение функций для решения задач**

*Цель*: научится применять функции для решения задач на языке программирования PHP.

**Задача.** Дан массив с числами. Создайте из него новый массив, где останутся лежать только положительные числа. Создайте для этого вспомогательную функцию **isPositive**, которая параметром будет принимать число и возвращать true, если число положительное, и false - если отрицательное.

**Решение:**

<?php

$arr **=** [1, 2, 3, **-**1, **-**2, **-**3];

function isPositive($num)

{if ($num **>=** 0) { return true;} else { return false;}}

$newArr **=** [];

foreach ($arr as $elem) {if (isPositive($elem)) {$newArr[] **=** $elem;}}

var\_dump($newArr);

?>

 Сделайте функцию **isNumberInRange**, которая параметром принимает число и проверяет, что оно больше нуля и меньше 10. Если это так - пусть функция возвращает **true**, если не так - **false**.

Решение:

Первый вариант:

<?php

function isNumberInRange($num)

{if ($num **>** 0 and $num **<** 10) { return true;} else { return false;} }

?>

Второй вариант - сокращенный:

<?php

function isNumberInRange($num)

{return $num **>** 0 and $num **<** 10; }

?>

1. Дан массив с числами. Запишите в новый массив только те числа, которые больше нуля и меньше 10-ти. Для этого используйте вспомогательную функцию **isNumberInRange** из предыдущей задачи.
2. Сделайте функцию **getDigitsSum** (digit - это цифра), которая параметром принимает целое число и возвращает сумму его цифр.
3. Найдите все года от 1 до 2019, сумма цифр которых равна **13**. Для этого используйте вспомогательную функцию **getDigitsSum** из предыдущей задачи.
4. Сделайте функцию **isEven()** (even - это четный), которая параметром принимает целое число и проверяет: четное оно или нет. Если четное - пусть функция возвращает **true**, если нечетное - **false**.
5. Дан массив с целыми числами. Создайте из него новый массив, где останутся лежать только четные из этих чисел. Для этого используйте вспомогательную функцию **isEven** из предыдущей задачи.
6. Сделайте функцию **getDivisors**, которая параметром принимает число и возвращает массив его делителей (чисел, на которое делится данное число).
7. Сделайте функцию **getCommonDivisors**, которая параметром принимает 2 числа, а возвращает массив их общих делителей. Для этого используйте вспомогательную функцию **getDivisors** из предыдущей задачи.

*Контрольные вопросы:*

1. Что такое функции?
2. Какие функции бывают?
3. Какие параметры бывают?

**Решение задач на заполнение и вывод линейного массива**

***Цель:***«Дать понятие одномерного массива, объяснить ввод массива, научиться заполнять массив данными различными способами; формирование навыков решения задач.

1. Заполните массив следующим образом: в первый элемент запишите 'x', во второй 'xx', в третий 'xxx' и так далее.

Решение:

<?php

$arr **=** [];

$str **=** 'х';

for ($i **=** 0; $i **<** 10; $i**++**) {

$arr[] **=** $str;

$str .**=** 'x';

}

var\_dump($arr);

?>

1. С помощью двух вложенных циклов заполните массив следующим образом: в первый элемент запишите '1', во второй '22', в третий '333' и так далее.

Решение:

<?php

$arr **=** [];

for ($i **=** 1; $i **<=** 9; $i**++**) {

$str **=** '';

for ($j **=** 0; $j **<** $i; $j**++**) {

$str .**=** $i;

}

$arr[] **=** $str;

}

var\_dump($arr);

?>

1. Сделайте функцию **arrayFill**, которая будет заполнять массив заданными значениями. Первым параметром функция принимает значение, которым заполнять массив, а вторым - сколько элементов должно быть в массиве. Пример: arrayFill('x', 5) сделает массив ['x', 'x', 'x', 'x', 'x'].
2. Дан массив с числами. Узнайте сколько элементов с начала массива надо сложить, чтобы в сумме получилось больше 10-ти. Считайте, что в массиве есть нужное количество элементов.

*Контрольные вопросы:*

1. Что такое массив?
2. Что такое индекс массива?
3. Как описать массив?
4. Как обрабатывать массив?
5. Как вводить массив?
6. Как выполнять действия с массивами?

**Решение задач на заполнение и вывод двумерного массива. Составление отчетной документации**

**Цель:** изучить структуру двумерного массива, научится работать с двумерными массивами, создать программы, выполняющие алгебраические операции с матрицами

* 1. Дан двухмерный массив с числами, например [[1, 2, 3], [4, 5], [6]]. Найдите сумму элементов этого массива. Массив, конечно же, может быть произвольным.

Решение:

<?php

$arr **=** [[1, 2, 3], [4, 5], [6,8]];

$sum **=** 0;

foreach ($arr as $elem) {

foreach($elem as $num) {

$sum **+=** $num;}}

echo $sum;

?>

* 1. Дан трехмерный массив с числами, например [[[1, 2], [3, 4]], [[5, 6], [7, 8]]]. Найдите сумму элементов этого массива. Массив, конечно же, может быть произвольным.
  2. С помощью двух циклов создайте массив [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
  3. Дана квадратная матрица размера n. Найти максимальный элемент побочной диагонали этой матрицы
  4. Даны целые положительные числа m и n. Сформировать целочисленную матрицу размера m×n, у которой все элементы имеют значения i‒j(i=1,2...m; j=1,2...n)
  5. Дана матрица размера m×n и целое положительное число k (1<k<m). Вывести элементы k-той строки этой матрицы
  6. Дана матрица размера m×n и целое положительное число k (1<k<n). Вывести элементы k-того столбца этой матрицы
  7. Дана матрица размера m×n и целое положительное число k (1<k<n). Вывести элементы k -того столбца этой матрицы
  8. Даны целые положительные числа m, n и k. Сформировать целочисленную матрицу размера m×n, у которой все элементы имеют значения k + i+ j (i=1,2...m; j=1,2...n)
  9. Даны целые положительные числа m, n и k. Сформировать целочисленную матрицу размера m×n , у которой все элементы имеют значения k‒(i+j)(i=1,2...m; j=1,2...n)

*Контрольные вопросы:*

1. Что такое двумерный массив?
2. Что такое индекс у двумерного массива?
3. Как описать двумерный массив?
4. Как обрабатывать массив?

**Решение задач на обработку строк**

*Цель:* изучить строковый тип. Изучить стандартные функции работы со строками, научиться выполнять стандартные операции со строками.

**Задача.** Дана строка **'minsk'**. Сделайте из нее строку **'MINSK'**.

**Решение:** воспользуемся [функцией strtoupper](http://code.mu/php/string/strtoupper.html) и сразу получим результат:

<?php

echo strtoupper('minsk');

?>

**Задача.** Дана строка **'минск'**. Сделайте из нее строку **'МИНСК'**.

**Решение:** функцией **strtoupper** мы не можем воспользоваться, так как она некорректно работает с русскими буквами. Воспользуемся [функцией mb\_strtoupper](http://code.mu/php/string/strtoupper.html) и сразу получим результат:

<?php

echo mb\_strtoupper('минск');

?>

**Задача.** Дана строка **'MINSK'**. Сделайте из нее строку **'Minsk'**.

**Решение:** готовой функции для решения задачи в PHP не существует. Поэтому сначала воспользуемся [функцией strtolower](http://code.mu/php/string/strtolower.html) (в результате получится **'minsk'**), а затем [функцией ucfirst](http://code.mu/php/string/ucfirst.html):

<?php

echo ucfirst(strtolower('minsk'));

?>

**Задача.** В переменной **$date** лежит дата в формате **'31-12-2030'**. Преобразуйте эту дату в формат **'2030.12.31'**.

**Решение:** для начала разобьем строку **'31-12-2030'** в массив с помощью [функции explode](http://code.mu/php/string/explode.html):

<?php

//Разбиваем строку в массив по разделителю '-':

$arr **=** explode('-', '31-12-2030');

?>

В полученном массиве в **$arr[0]** будет лежать **31** (то есть день), в **$arr[1]** - месяц, а в **$arr[2]** - год. Сольем элементы этого массива в новую строку:

<?php

//Разбиваем строку в массив по разделителю '-':

$arr **=** explode('-', '31-12-2030');

//Получим дату в нужном формате:

echo $arr[2].'.'.$arr[1].'.'.$arr[0];

?>

### Работа с регистром символов

*Для решения задач данного блока вам понадобятся следующие функции:*[*strtolower*](http://code.mu/php/string/strtolower.html)*,*[*strtoupper*](http://code.mu/php/string/strtoupper.html)*,*[*ucfirst*](http://code.mu/php/string/ucfirst.html)*,*[*lcfirst*](http://code.mu/php/string/lcfirst.html)*,*[*ucwords*](http://code.mu/php/string/ucwords.html)*.*

1. Дана строка **'php'**. Сделайте из нее строку **'PHP'**.
2. Дана строка **'PHP'**. Сделайте из нее строку **'php'**.
3. Дана строка **'london'**. Сделайте из нее строку **'London'**.
4. Дана строка **'London'**. Сделайте из нее строку **'london'**.
5. Дана строка **'london is the capital of great britain'**. Сделайте из нее строку **'London Is The Capital Of Great Britain'**.
6. Дана строка **'LONDON'**. Сделайте из нее строку **'London'**

### Работа с strlen

*Для решения задач данного блока вам понадобятся следующие функции:*[*strlen*](http://code.mu/php/string/strlen.html)*.*

1. Дана строка **'html css php'**. Найдите количество символов в этой строке.
2. Дана переменная **$password**, в которой хранится пароль пользователя. Если количество символов пароля больше 5-ти и меньше 10-ти, то выведите пользователю сообщение о том, что пароль подходит, иначе сообщение о том, что нужно придумать другой пароль.

### Работа с substr

*Для решения задач данного блока вам понадобятся следующие функции:*[*substr*](http://code.mu/php/string/substr.html)*.*

1. Дана строка **'html css php'**. Вырежьте из нее и выведите на экран слово 'html', слово 'css' и слово 'php'.
2. Дана строка. Вырежите и выведите на экран последние **3** символа этой строки.
3. Дана строка. Проверьте, что она начинается на **'http://'**. Если это так, выведите 'да', если не так - 'нет'.
4. Дана строка. Проверьте, что она начинается на **'http://'** или на **'https://'**. Если это так, выведите 'да', если не так - 'нет'.
5. Дана строка. Проверьте, что она заканчивается на **'.png'**. Если это так, выведите 'да', если не так - 'нет'.
6. Дана строка. Проверьте, что она заканчивается на **'.png'** или на **'.jpg'**. Если это так, выведите 'да', если не так - 'нет'.
7. Дана строка. Если в этой строке более 5-ти символов - вырежите из нее первые **5** символов, добавьте троеточие в конец и выведите на экран. Если же в этой строке **5** и менее символов - просто выведите эту строку на экран.

### Работа с str\_replace

*Для решения задач данного блока вам понадобятся следующие функции:*[*str\_replace*](http://code.mu/php/string/str_replace.html)*.*

1. Дана строка **'31.12.2013'**. Замените все точки на дефисы.
2. Дана строка **$str**. Замените в ней все буквы 'a' на цифру 1, буквы 'b' - на 2, а буквы 'c' - на 3.
3. Дана строка с буквами и цифрами, например, '**1a2b3c4b5d6e7f8g9h0**'. Удалите из нее все цифры. То есть в нашем случае должна получится строка '**abcbdefgh**'.

### Работа с strtr

*Для решения задач данного блока вам понадобятся следующие функции:*[*strtr*](http://code.mu/php/string/strtr.html)*.*

1. Дана строка **$str**. Замените в ней все буквы 'a' на цифру 1, буквы 'b' - на 2, а буквы 'c' - на 3. Решите задачу двумя способами работы с функцией **strtr** (массив замен и две строки замен)

### Работа с substr\_replace

*Для решения задач данного блока вам понадобятся следующие функции:*[*substr\_replace*](http://code.mu/php/string/substr_replace.html)*.*

1. Дана строка **$str**. Вырежите из нее подстроку с 3-го символа (отсчет с нуля), 5 штук и вместо нее вставьте '!!!'.

### Работа с strpos, strrpos

*Для решения задач данного блока вам понадобятся следующие функции:*[*strpos*](http://code.mu/php/string/strpos.html)*,*[*strrpos*](http://code.mu/php/string/strrpos.html)*.*

1. Дана строка 'abc abc abc'. Определите позицию первой буквы 'b'.
2. Дана строка 'abc abc abc'. Определите позицию последней буквы 'b'.
3. Дана строка 'abc abc abc'. Определите позицию первой найденной буквы 'b', если начать поиск не с начала строки, а с позиции 3.
4. Дана строка 'aaa aaa aaa aaa aaa'. Определите позицию второго пробела.
5. Проверьте, что в строке есть две точки подряд. Если это так - выведите 'есть', если не так - 'нет'.
6. Проверьте, что строка начинается на **'http://'**. Если это так - выведите 'да', если не так - 'нет'.

### Работа с explode, implode

*Для решения задач данного блока вам понадобятся следующие функции:*[*explode*](http://code.mu/php/string/explode.html)*,*[*implode*](http://code.mu/php/array/implode.html)*.*

1. Дана строка **'html css php'**. С помощью [функции explode](http://code.mu/php/string/explode.html) запишите каждое слово этой строки в отдельный элемент массива.
2. Дан массив с элементами **'html', 'css', 'php'**. С помощью [функции implode](http://code.mu/php/array/implode.html) создайте строку из этих элементов, разделенных запятыми.
3. В переменной **$date** лежит дата в формате **'2013-12-31'**. Преобразуйте эту дату в формат **'31.12.2013'**.

### Работа с str\_split

*Для решения задач данного блока вам понадобятся следующие функции:*[*str\_split*](http://code.mu/php/string/str_split.html)*.*

1. Дана строка **'1234567890'**. Разбейте ее на массив с элементами **'12', '34', '56', '78', '90'**.
2. Дана строка **'1234567890'**. Разбейте ее на массив с элементами **'1', '2', '3', '4', '5', '6', '7', '8', '9', '0'**.
3. Дана строка **'1234567890'**. Сделайте из нее строку **'12-34-56-78-90'** не используя цикл.

### Работа с trim, ltrim, rtrim

*Для решения задач данного блока вам понадобятся следующие функции:*[*trim*](http://code.mu/php/string/trim.html)*,*[*ltrim*](http://code.mu/php/string/ltrim.html)*,*[*rtrim*](http://code.mu/php/string/rtrim.html)*.*

1. Дана строка. Очистите ее от концевых пробелов.
2. Дана строка **'/php/'**. Сделайте из нее строку **'php'**, удалив концевые слеши.
3. Дана строка **'слова слова слова.'**. В конце этой строки может быть точка, а может и не быть. Сделайте так, чтобы в конце этой строки гарантировано стояла точка. То есть: если этой точки нет - ее надо добавить, а если есть - ничего не делать. Задачу решите через **rtrim** без всяких ифов.

### Работа с strrev

*Для решения задач данного блока вам понадобятся следующие функции:*[*strrev*](http://code.mu/php/string/strrev.html)*.*

1. Дана строка **'12345'**. Сделайте из нее строку **'54321'**.
2. Проверьте, является ли слово **палиндромом** (одинаково читается во всех направлениях, примеры таких слов: **madam, otto, kayak, nun, level**).

### Работа с str\_shuffle

*Для решения задач данного блока вам понадобятся следующие функции:*[*str\_shuffle*](http://code.mu/php/string/str_shuffle.html)*.*

1. Дана строка. Перемешайте символы этой строки в случайном порядке. [.](http://code.mu/tasks/php/base/rabota-so-strokovymi-funkciyami-v-php.html)
2. Создайте строку из 6-ти случайных маленьких латинских букв так, чтобы буквы не повторялись. Нужно сделать так, чтобы в нашей строке могла быть любая латинская буква, а не ограниченный набор.

### Работа с number\_format

*Для решения задач данного блока вам понадобятся следующие функции:*[*number\_format*](http://code.mu/php/string/number_format.html)*.*

1. Дана строка **'12345678'**. Сделайте из нее строку **'12 345 678'**.

### Работа с str\_repeat

*Для решения задач данного блока вам понадобятся следующие функции:*[*str\_repeat*](http://code.mu/php/string/str_repeat.html)*.*

1. Нарисуйте пирамиду, как показано на рисунке, только у вашей пирамиды должно быть 9 рядов, а не 5. Решите задачу с помощью **одного цикла** и [функции str\_repeat](http://code.mu/php/string/str_repeat.html).

x  
xx  
xxx  
xxxx  
xxxxx

1. Нарисуйте пирамиду, как показано на рисунке. Решите задачу с помощью **одного цикла** и [функции str\_repeat](http://code.mu/php/string/str_repeat.html).

1  
22  
333  
4444  
55555  
666666  
7777777  
88888888  
999999999

### Работа с strip\_tags и htmlspecialchars

*Для решения задач данного блока вам понадобятся следующие функции:*[*htmlspecialchars*](http://code.mu/php/string/htmlspecialchars.html)*,*[*strip\_tags*](http://code.mu/php/string/strip_tags.html)*.*

1. Дана строка **'html, <b>php</b>, js'**. Удалите теги из этой строки.
2. Дана строка **$str**. Удалите все теги из этой строки, кроме тегов <b> и <i>.
3. Дана строка **'html, <b>php</b>, js'**. Выведите ее на экран **'как есть'**: то есть браузер не должен преобразовать <b> в жирный.

### Работа с chr и ord

*Для решения задач данного блока вам понадобятся следующие функции:*[*chr*](http://code.mu/php/string/chr.html)*,*[*ord*](http://code.mu/php/string/ord.html)*.*

1. Узнайте код символов **'a', 'b', 'c', пробела**.  Изучите [таблицу ASCII](http://www.asciitable.com/). Определите границы, в которых располагаются буквы английского алфавита.
2. Выведите на экран символ с кодом **33**.
3. Запишите в переменную **$str** случайный символ - большую букву латинского алфавита.*Подсказка: с помощью таблицы ASCII определите какие целые числа соответствуют большим буквам латинского алфавита*
4. Запишите в переменную **$str** случайную строку **$len** длиной, состоящую из маленьких букв латинского алфавита. *Подсказка: воспользуйтесь циклом for или while.*
5. Дана буква английского алфавита. Узнайте, она маленькая или большая.

### Работа с strchr, strrchr

*Для решения задач данного блока вам понадобятся следующие функции:*[*strchr*](http://code.mu/php/string/strchr.html)*,*[*strrchr*](http://code.mu/php/string/strrchr.html)*.*

1. Дана строка **'ab-cd-ef'**. С помощью функции **strchr** выведите на экран строку **'-cd-ef'**.
2. Дана строка **'ab-cd-ef'**. С помощью функции **strrchr** выведите на экран строку **'-ef'**.

### Работа с strstr

*Для решения задач данного блока вам понадобятся следующие функции:*[*strstr*](http://code.mu/php/string/strstr.html)*.*

1. Дана строка **'ab--cd--ef'**. С помощью функции **strstr** выведите на экран строку **'--cd--ef'**.
2. Преобразуйте строку **'var\_test\_text'** в **'varTestText'**. Скрипт, конечно же, должен работать с любыми аналогичными строками.
3. Дана строка следующего вида: 5 цифр, затем пробел, затем еще 5 цифр. Например, дана такая строка **'12345 67890'**. Сделайте из нее строку **'54321 09876'**.
4. Дан массив с числами. Выведите на экран все числа, в которых есть цифра 3.
5. Дан массив с числами. Посчитайте суммарное количество цифр 3 в этих числах.

*Контрольные вопросы:*

1. Что такое строка?
2. Чему равна длина строки ?
3. Операции над строками?
4. Стандартные функции работы со строками
5. Стандартные процедуры для работы со строками

**Ввод данных методом GET, POST при помощи форм. Обработка GETи POST массивов**

*Цель:* Получение практических навыков по передаче значений переменным в сценариях РНР с использованием формы

**Использование HTML-форм для передачи данных на сервер**

Для передачи данных на сервер в языке HTML есть специальная конструкция – формы. Формы предназначены для того чтобы получать от пользователя информацию. Например, вам нужно знать логин и пароль пользователя для того, чтобы определить, на какие страницы сайта его можно допускать. Или вам необходимы личные данные пользователя, чтобы была возможность с ним связаться. Формы как раз и применяются для ввода такой информации. В них можно вводить текст или выбирать подходящие варианты из списка. Данные, записанные в форму, отправляются для обработки специальной программе (например, скрипту на PHP) на сервере. В зависимости от введенных пользователем данных эта программа может формировать различные web-страницы, отправлять запросы к базе данных, запускать различные приложения и т.п.

Для создания формы в языке HTML используется тег FORM. Внутри него находится одна или несколько команд INPUT. С помощью атрибутов ACTION и METHOD тега FORM задаются имя программы, которая будет обрабатывать данные формы, и метод запроса, соответственно. Команда INPUT определяет тип и различные характеристики запрашиваемой информации. Отправка данных формы происходит после нажатия кнопки

INPUT типа SUBMIT

|  |  |
| --- | --- |
| <form action="1.php" method=POST> | Данные формы будет обрабатывать файл |
| ……… | 1.php, при отправке запроса будет использован |
| ……… |
| метод POST. |
| ……… |
| </form> |  |
|  |  |

При отправке данных формы с помощью метода GET содержимое формы добавляется к URL после знака вопроса в виде пар имя=значения, объединенных с помощью амперсанта &:

action?name1=value1&name2=value2&name3=value3

Здесь action – это URL-адрес программы, которая должна обрабатывать форму (это либо программа, заданная в атрибуте action тега form, либо сама текущая программа, если этот атрибут опущен). Имена name1, name2, name3 соответствуют именам элементов формы, а value1, value2, value3 – значениям этих элементов. Все специальные символы, включая = и &, в именах или значениях этих параметров будут опущены. Поэтому не стоит использовать в названиях или значениях элементов формы эти символы и символы кириллицы в идентификаторах.

Для кнопок типа checkbox и radiobutton значение value определяется атрибутом VALUE в том случае, когда кнопка отмечена. Не отмеченные кнопки при составлении строки запроса игнорируются целиком. Несколько кнопок типа checkbox могут иметь один атрибут NAME (и различные VALUE), если это необходимо. Кнопки типа radiobutton предназначены для одного из всех предложенных вариантов и поэтому должны иметь одинаковый атрибут NAME и различные атрибуты VALUE.

Для метода POST содержимое формы кодируется точно так же, как для метода GET, но вместо добавления строки к URL содержимое запроса посылается блоком данных как часть операции POST. Если присутствует атрибут ACTION, то значение URL, которое там находится, определяет, куда посылать этот блок данных. Этот метод рекомендуется для передачи больших по объему блоков данных.

Передать данные методом POST можно только с помощью HTML-формы, поскольку данные передаются в теле запроса, а не в заголовке, как в GET. Соответственно и изменить значение параметров можно, только изменив значение, введенное в форму. При использовании POST пользователь не видит передаваемые серверу данные.

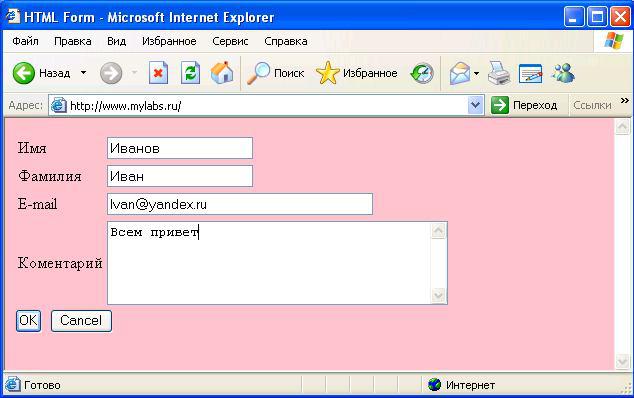
Основное преимущество POST запросов – это их большая безопасность и функциональность по сравнению с GET-запросами. Поэтому метод POST чаще используют для передачи важной информации, а также информации большого объема

**ПРАКТИЧЕСКАЯ ЧАСТЬ**

1. Рассмотрим пример создания формы (index.html), в которую пользователем вводятся данные:

Имя; Фамилия;

Адрса электронной почты; Текст комментария.



index.html

<html>

<head>

<title> HTML Form </title>

</head>

<body bgcolor=pink>

**<form action="handleform.php" method=POST>**

<table>

<tr>

<td> Имя</td>

<td><input type=text name="FirstName" size=20></td>

</tr><tr><td>Фамилия</td>

<td><input type=text name="LastName" size=20></td>

</tr><tr><td>E-mail</td>

<td><input type=text name="Email" size=40></td></tr><tr>

<td> Комментарий</td>

<td><textarea name="Comments" rows=5 cols=40></textarea></td> </tr>

</table>

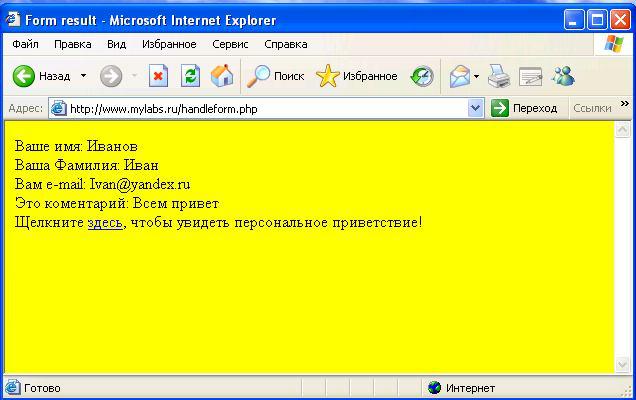
<input type=submit name="Submit" value="OK">&nbsp; <input type=reset name="Reset" value="Cancel"> </form>

</body>

</html>

Затем PHP-сценарий (рhandleform.php) получаетя данные с формы и

отображает извлеченные из формы данные в окне браузера.



handleform.php

<html>

<head>

<title> Form result</title>

</head>

<body bgcolor=yellow>

<?PHP

print("Ваше имя: $\_POST[FirstName] <br>");

print("Ваша Фамилия: $\_POST[LastName] <br>");

print("Вам e-mail: $\_POST[Email] <br>");

print("Это комментарий: $\_POST[Comments] <br>");

print("Щелкните <A

href=myhello.php?FirstName=$\_POST[FirstName]&Lastname=$\_POST[LastName]>зд

есь</a>, чтобы увидеть персональное приветствие!");

?>

</body></html>

Далее сценарий при щелчке на ссылке, генерирует отправку на еще один

PHP-файл (myhello.php), который отображает персональное приветствие

пользователю.

myhello.php

<html>

<head>

<title>Greetings!!!!</title>

</head>

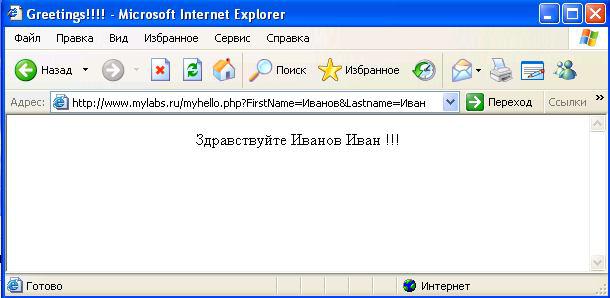
<body>

<?PHP

print("<center> Здравствуйте $\_GET[FirstName] $\_GET[Lastname] !!!</center>");

?></body>

</html>



Поменяйте в файле index.html значение атрибута method c POST на GET и посмотрите что изменилось.

1. Рассмотрим создание формы и РНР-сценария в одном файле

anketa.php

<head>

<title> Anketa</title>

</head>

<body bgcolor=yellow>

<?PHP

**if (!isset($\_POST[submit]))**

{print("<form action=\"anketa.php\" method=POST>"); print("<table> <tr><td> Имя</td>");

print("<td><input type=text name=\"FirstName\" size=20></td></tr>"); print("<tr><td>Фамилия</td>");

print("<td><input type=text name=\"LastName\" size=20></td></tr>"); print("<tr><td>E-mail</td>");

print("<td><input type=text name=\"Email\" size=40></td></tr>"); print("<tr><td> Комментарий</td>");

print("<td><textarea name=\"Comments\" rows=5 cols=40></textarea></td></tr></table>");

print("<input type=submit name=\"submit\" value=\"OK\">&nbsp;");

print("<input type=reset name=\"Reset\" value=\"Cancel\">");

print("</form>");}

else{$FirstName=**trim**($\_POST[FirstName]);

$Lastname=**trim**($\_POST [LastName]);

$Email=**trim**($\_POST [Email]);

$Comments=trim($\_POST[Comments]);

$Name=$\_POST[FirstName]." ".$\_POST[LastName];

print("Ваше имя: $FirstName <br>");

print("Ваша фамилия: $Lastname <br>");

print("Ваш e-mail is: $Email <br>");

print("Комментарий: $Comments <br>");

print("Щелкните <A href=\"myhello.php?Name=$Name\">здесь</a>, чтобы увидеть

персональное приветствие!");}?>

</body>

</html>

Отредактируйте файл myhello.php следующим образом

myhello.php

<html>

<head>

<title>Greetings!!!!</title>

</head>

<body>

<?PHP

print("<center> Здравствуйте $\_GET[Name] !!!</center>");

?>

</body>

</html>

***Примечание:***

1. Функция trim *–принимает в качестве своего единственного* *аргумента строку, и удаляет из нее пробелы слева и справа.*
2. Функция isset *–определяет,* *установлена ли переменная.*

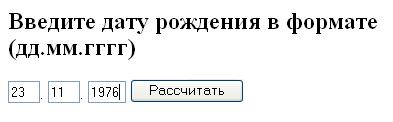
*Синтаксис*

bool isset ( $var )

*Возвращает TRUE, если* var *существует, в противном случае возвращается FALSE.*

**ЗАДАНИЕ**

1. Создайте форму для ввода даты рождения и напишите скрипт для вычисления возраста и вывода его на экран (используйте метод GET).

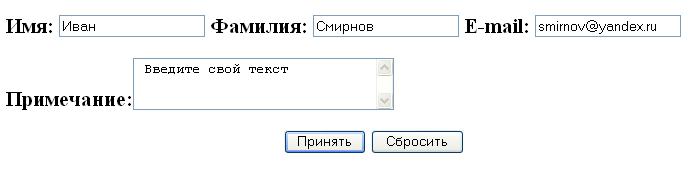


***Примечание:*** *Для определения текущей даты используйте функцию*date(“m.d.Y”)*.*

*Чтобы выделить число, месяц и год допускается следующее использование:*

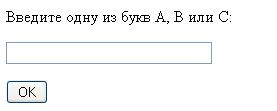
date(“d”) *–* *возвращает число текущего месяца;* date(“m”) *–* *возвращает номер месяца текущего года;* date(“Y”) *–* *возвращает текущий год.*

1. Разработайте форму как показано ниже



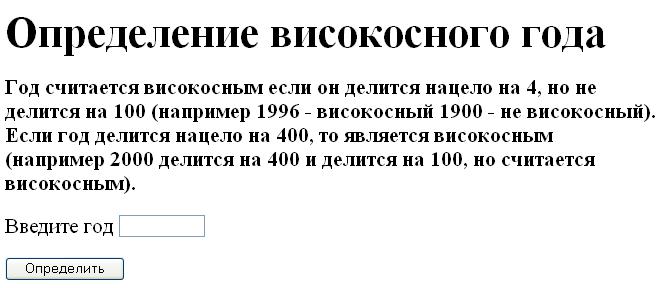
При нажатии на кнопку **Принять**, если поля **Имя** и **Фамилия** не пустые, то содержимое формы должно отображаться в новом окне в противном случае должно появиться окно о необходимости заполнить соответствующие поля (используйте метод POST).

1. Создайте форму

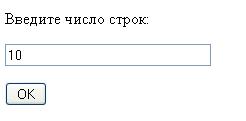


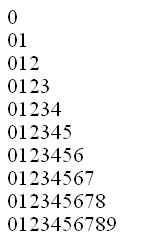
напишите скрипт, который, в зависимости от введенной буквы выводит текст: "Вы ввели букву А" или "Вы ввели букву B" или "Вы ввели букву С".

1. Создайте форму как показано ниже и напишите скрипт определяющий по нажатию на кнопку **Определить** является ли введенный год високосным.



1. Создайте форму вида

и напишите скрипт выводящий числа в виде:



Задание:

1. Отправьте с помощью GET-запроса число. Выведите его на экран.
2. Отправьте с помощью GET-запроса число. Выведите его на экран квадрат этого числа.
3. Отправьте с помощью GET-запроса два числа. Выведите его на экран сумму этих чисел.
4. Пусть с помощью GET-запроса отправляется число. Оно может быть или 1, или 2. Сделайте так, чтобы если передано 1 - на экран вывелось слово 'привет', а если 2 - то слово 'пока'.
5. Дан массив. Сделайте так, чтобы с помощью GET-запроса можно было вывести любой элемент этого массива.
6. Сделайте 3 ссылки. Пусть первая передает число 1, вторая - число 2, третья - число 3. Сделайте так, чтобы по нажатию на ссылку на экран выводилось ее число.
7. Сформируйте в цикле 10 ссылок. Пусть каждая ссылка передает свое число. Сделайте так, чтобы по нажатию на ссылку на экран выводилось ее число.
8. Модифицируйте предыдущую задачу так, чтобы каждая ссылка стояла в своем абзаце.
9. Модифицируйте предыдущую задачу так, чтобы каждая ссылка стояла в своем **li** в теге **ul**.
10. Дан массив. Сделайте так, чтобы с помощью GET-запроса можно было вывести любой элемент этого массива. Для этого вручную сделайте ссылку для каждого элемента массива. Переходя любой ссылке мы должны увидеть на экране соответствующий элемент массива.
11. Модифицируйте предыдущую задачу так, чтобы ссылки выводились с помощью цикла **foreach**автоматически в соответствии с количеством элементов в массиве.

**Вывод серверных переменных**

*Цель:* Научится создавать и выводить данные из базы данных на страницу.

*Теоретические сведенья*

**Устанавливаем соединение с БД**

Прежде, чем начать работать с базой данных из PHP, нужно **установить соединение с сервером**, на котором эта база находится.

Делается это с помощью функции PHP **mysql\_connect**, которая принимает 3 параметра: **имя хоста**(сервера), **имя пользователя**, под которым мы работаем с базой и **пароль** для этого пользователя.

Если вы работаете на своем компьютере то это будут **'localhost'**, **'root'** и пароль в виде пустой строки (на некоторых серверах он тоже может быть **root**). Если ваша база данных **в интернете** - то эти данные дает вам **хостер**.

Давайте установим соединение с базой данных:

<?php

//Устанавливаем доступы к базе данных:

$host **=** 'localhost'; //имя хоста, на локальном компьютере это localhost

$user **=** 'root'; //имя пользователя, по умолчанию это root

$password **=** ''; //пароль, по умолчанию пустой

$db\_name **=** 'test'; //имя базы данных

//Соединяемся с базой данных используя наши доступы:

$link **=** mysqli\_connect($host, $user, $password, $db\_name);

/\*

Соединение записывается в переменную $link,

которая используется дальше для работы mysqi\_query.

\*/

?>

**Посылаем запросы к базе данных**

**Запросы** к базе данных представляют собой обычные строки, которые мы вставляем в функцию PHP**mysqli\_query** (первым параметром это функция принимает переменную, в которую мы записали результат mysqli\_connect, в нашем случае это переменная $link):

<?php

//Внутри функции PHP mysqli\_query стоит обычная строка:

mysqli\_query($link, "SELECT\*FROM workers WHERE id>0");

//Мы можем формировать эту строку с помощью переменных:

$table **=** 'workers'; //задаем имя таблицы в переменной

mysqli\_query($link, "SELECT\*FROM ".$table." WHERE id>0");

?>

Принято правило, по которому *команды SQL следует писать в верхнем регистре* (то есть большими буквами), а все остальное - в нижнем.

Это относиться к командам **SELECT, UPDATE, FROM, DELETE, WHERE** и другим такого рода.

Конечно, синтаксической **ошибки не будет**, если вы напишите их маленькими буквами, но принято большими.

**Отлавливаем ошибки базы данных**

Многие начинающие зачастую не умеют **отлавливать ошибки**, которые вернула база данных.

Поэтому при работе с БД у них постоянно возникают сложности. Что-то не работает, а что - не понятно, так как ошибок они не видят, так как PHP не выводит ошибки mySQL, если ему об этом не сказать

Чтобы вывести ошибки, следует пользоваться конструкцией **or die ( mysqli\_error($link) )**, которую необходимо добавлять к каждому запросу к БД.

Смотрите пример: **mysqli\_query($link, $query) or die( mysqli\_error($link) );**

Таким образом вы сразу будете получать сообщения об ошибках синтаксиса **SQL**. Обратите внимание на то, что на рабочем сайте эти конструкции следует удалять, чтобы пользователи и тем более хакеры не видели ошибок БД.

**Проблемы с кодировками**

Зачастую у новичков возникает **проблема с кодировками** - казалось бы нормальный русский текст в базу данных записывается абракадаброй или вопросиками.

Чтобы не было проблем с кодировками следует придерживаться простых правил:

* Базу данных следует создавать в кодировке utf8\_general\_ci.
* Документ PHP должен быть в кодировке utf8.
* Таблицы в БД должны быть в utf8\_general\_ci.
* На всякий случай сразу после команды mysqli\_connect добавьте такое запрос: mysqli\_query($link, "SET NAMES 'utf8'");

**Начнем практиковаться**

Сейчас мы с вами начнем изучить SQL запросы на практике. Для этого нам понадобится тестовая таблица в базе данных, заполненная некоторыми данными. Сейчас мы с вами ее сделаем и заполним.

Итак, создайте свою первую базу данных с помощью PhpMyAdmin.

Назовите ее **"test"**.

Создайте в этой базе новую таблицу.

Назовите ее **"workers"** (англ. работники).

В ней создайте 4 столбца (столбцы по другому называются **поля**):

* **id** – тип integer, не забудьте поставить ему галочку AUTO\_INCREMENT (чтобы в этом столбце номера проставлялись автоматически).
* **name** (англ. имя) – тип varchar, размером в 256 знаков.
* **age** (англ. возраст) - тип integer.
* **salary** (англ. зарплата) - тип integer.

Ее заполните тестовыми данными, как показано в таблице ниже (этот шаг обязателен, так как дальше все задачи будут по этой таблице):

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **name** | **age** | **salary** |
| 1 | Дима | 23 | 400 |
| 2 | Петя | 25 | 500 |
| 3 | Вася | 23 | 500 |
| 4 | Коля | 30 | 1000 |
| 5 | Иван | 27 | 500 |
| 6 | Кирилл | 28 | 1000 |

Итак, у нас есть таблица с работниками фирмы, в которой указаны их имена, возрасты и зарплаты (в $). Далее мы будем работать с этой таблицей.

**Тестируем работоспособность**

Прежде чем начать работу, следует протестировать работоспособность - попробуем сделать какой-нибудь запрос к нашей базе.

Просто скопируйте этот код и запустите его у себя:

<?php

//Устанавливаем доступы к базе данных:

$host **=** 'localhost'; //имя хоста, на локальном компьютере это localhost

$user **=** 'root'; //имя пользователя, по умолчанию это root

$password **=** ''; //пароль, по умолчанию пустой

$db\_name **=** 'test'; //имя базы данных

//Соединяемся с базой данных используя наши доступы:

mysqli\_connect($host, $user, $password, $db\_name) or die(mysqli\_error($link));

//Устанавливаем кодировку (не обязательно, но поможет избежать проблем):

mysqli\_query($link, "SET NAMES 'utf8'");

//Формируем тестовый запрос:

$query **=** "SELECT \* FROM workers WHERE id > 0";

//Делаем запрос к БД, результат запроса пишем в $result:

$result **=** mysqli\_query($link, $query) or die(mysqli\_error($link));

//Проверяем что же нам отдала база данных, если null – то какие-то проблемы:

var\_dump($result);?>

Если **var\_dump($result)** вернет **resource**, то все работает, если же **null** – то возникли какие-то проблемы.

В таком случае проверьте все еще раз, уберите последовательно все ошибки **PHP**, если таковые есть.

**Как достать результат**

После того, как мы сделали запрос к базе, в переменной **$result** будет лежать результат этого действия.

Однако лежит он не в той форме, которая нам нужна в PHP, а в той форме, в которой его прислала нам база.

Достать результат в нормальном виде (в массиве) можно с помощью следующего кода:

<?php

//Делаем запрос к БД, результат запроса пишем в $result:

$result **=** mysqli\_query($link, $query) or die(mysqli\_error($link));

//Преобразуем то, что отдала нам база в нормальный массив PHP $data:

for ($data **=** []; $row **=** mysqli\_fetch\_assoc($result); $data[] **=** $row);?>

Как работает последняя строка?

Функция **mysqli\_fetch\_assoc** считывает последовательно каждую строку результата, который прислала нам база.

В цикле **for** мы считываем построчно результат из базы.

Когда цикл дойдет до последней строки - **mysqli\_fetch\_assoc** вернет **false** и цикл **for** закончит свою работу.

*Практическое задание*

Все задачи будут по данной таблице **workers** (если не сказано иное):

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **name** | **age** | **salary** |
| 1 | Дима | 23 | 400 |
| 2 | Петя | 25 | 500 |
| 3 | Вася | 23 | 500 |
| 4 | Коля | 30 | 1000 |
| 5 | Иван | 27 | 500 |
| 6 | Кирилл | 28 | 1000 |

**Примеры решения задач**

**Задача**

**Задача.** Выбрать работника с id=10.

**Решение:**

<?php

$query **=** "SELECT \* FROM workers WHERE id=10";

?>

**Задача**

**Задача.** Выбрать работников с зарплатой 500$.

**Решение:**

<?php

$query **=** "SELECT \* FROM workers WHERE salary=500";

?>

**Задача**

**Задача.** Выбрать работников с зарплатой 500$ и id больше 3.

**Решение:**

<?php

$query **=** "SELECT \* FROM workers WHERE salary=500 AND id>3";

?>

**Задача**

**Задача.** Добавьте нового работника Джона, 20 лет, зарплата 700$.

**Решение:**

Воспользуемся первым синтаксисом:

<?php

$query **=** "INSERT INTO workers SET name='Джон', age=20, salary=700";

?>

Воспользуемся вторым синтаксисом:

<?php

$query **=** "INSERT INTO workers (name, age, salary) VALUES ('Джон', 20, 700)";

?>

**Задача**

**Задача.** Добавьте одним запросом трех новых работников: Катю, 20 лет, зарплата 500$, Юлю, 25 лет, зарплата 600$, Женю, 30 лет, зарплата 900$.

**Решение:** запрос должен выглядеть так:

INSERT INTO workers (name, age, salary)

VALUES ('Катя', 20, 500), ('Юля', 25, 600), ('Женя', 30, 900)

**Задача**

**Задача.** Удалите работника Джона.

**Решение:**

<?php

$query **=** "DELETE FROM workers WHERE name='Джон'";

?>

**Задача**

**Задача.** Поставьте Диме зарплату в 1000$.

**Решение:**

<?php

$query **=** "UPDATE workers SET salary=1000 WHERE name='Дима'";

?>

**Задача**

**Задача.** Поставьте Диме зарплату в 1000$ и возраст 20 лет.

**Решение:**

<?php

$query **=** "UPDATE workers SET salary=1000, age=20 WHERE name='Дима'";

?>

**На SELECT**

*Для решения задач данного блока вам понадобятся следующие SQL команды:*[*SELECT*](http://theory.phphtml.net/sql/select.html)*,*[*WHERE*](http://theory.phphtml.net/sql/where.html)*.*

 Выбрать работника с id = 3.

 Выбрать работников с зарплатой 1000$.

 Выбрать работников в возрасте 23 года.

 Выбрать работников с зарплатой более 400$.

 Выбрать работников с зарплатой равной или большей 500$.

 Выбрать работников с зарплатой НЕ равной 500$.

 Выбрать работников с зарплатой равной или меньшей 900$.

 Узнайте зарплату и возраст Васи.

**На OR и AND**

*Для решения задач данного блока вам понадобятся следующие SQL команды:*[*SELECT*](http://theory.phphtml.net/sql/select.html)*,*[*WHERE*](http://theory.phphtml.net/sql/where.html)*,*[*OR*](http://theory.phphtml.net/sql/or.html)*,*[*AND*](http://theory.phphtml.net/sql/and.html)*.*

Выбрать работников в возрасте от 25 (не включительно) до 28 лет (включительно).

 Выбрать работника Петю.

 Выбрать работников Петю и Васю.

 Выбрать всех, кроме работника Петя.

 Выбрать всех работников в возрасте 27 лет или с зарплатой 1000$.

 Выбрать всех работников в возрасте от 23 лет (включительно) до 27 лет (не включительно) или с зарплатой 1000$.

 Выбрать всех работников в возрасте от 23 лет до 27 лет или с зарплатой от 400$ до 1000$.

 Выбрать всех работников в возрасте 27 лет или с зарплатой не равной 400$.

**На INSERT**

*Для решения задач данного блока вам понадобятся следующие SQL команды:*[*INSERT*](http://theory.phphtml.net/sql/insert.html)*.*

 Добавьте нового работника Никиту, 26 лет, зарплата 300$. Воспользуйтесь **первым** синтаксисом.

 Добавьте нового работника Светлану с зарплатой 1200$. Воспользуйтесь **вторым** синтаксисом.

 Добавьте двух новых работников одним запросом: Ярослава с зарплатой 1200$ и возрастом 30, Петра с зарплатой 1000$ и возрастом 31.

**На DELETE**

*Для решения задач данного блока вам понадобятся следующие SQL команды:*[*DELETE*](http://theory.phphtml.net/sql/delete.html)*.*

Удалите работника с id=7.

 Удалите Колю.

 Удалите всех работников, у которых возраст 23 года.

**Верните таблицу workers в исходное состояние.**

**На UPDATE**

*Для решения задач данного блока вам понадобятся следующие SQL команды:*[*UPDATE*](http://theory.phphtml.net/sql/update.html)*.*

 Поставьте Васе зарплату в **200$**.

 Работнику с id=4 поставьте возраст **35** лет.

 Всем, у кого зарплата 500$ сделайте ее **700$**.

 Работникам с id больше 2 и меньше 5 включительно поставьте возраст **23**.

 Поменяйте Васю на Женю и прибавьте ему зарплату до **900$**.

**Подключение скриптов при помощи include(), require()**

**Цель** — освоить основные принципы структуризации и оптимизации кода PHP-скриптов.

**Пример 1.** Пусть в файле c у нас хранится набор каких-то параметров и функций. Каждый раз, когда нам нужно будет использовать эти параметры (функции), мы будем вставлять в текст нашей основной программы команду include 'c'.

c

<?php

$user = "Вася";

$today = date("d.m.y");

/\* функция date() возвращает дату

и время (здесь – дату в формате

день.месяц.год) \*/

?>

include.php

<?php

include ("c");

/\* переменные $user и $today заданы в файле

c. Здесь мы тоже можем ими

пользоваться благодаря команде

include("c") \*/

echo "Привет, $user!<br>";

// выведет "Привет, Вася!"

echo "Сегодня $today";

// выведет, например, "Сегодня 7.07.05"

?>

**Пример 2. Использование оператора включения include (,** **)**

Заметим, что использование оператора *include* эквивалентно простой вставке содержательной части файла c в код программы include.php. В момент вставки файла происходит переключение из режима обработки PHP в режим HTML. Поэтому код внутри включаемого файла, который нужно обработать как PHP-скрипт, должен быть заключен в соответствующие теги.

Поиск файла для вставки происходит по следующим правилам.

1. Сначала ведется поиск файла в *include\_path* относительно текущей рабочей директории.
2. Если файл не найден, то поиск производится в *include\_path* относительно директории текущего скрипта.
3. Параметр *include\_path*, определяемый в файле настроек PHP, задает имена директорий, в которых нужно искать включаемые файлы.

Пусть файл для вставки c останется таким же, а include.php будет следующим:

<?php

function Footer(){

// объявляем функцию с именем Footer

include ("c");

/\* включаем файл c.

Теперь его переменными можно пользоваться,

но только внутри функции \*/

$str = "Сегодня: $today <br>";

$str .= "<a

href='mailto:help@'>Страницу

создал $user</a>";

echo "$str";

}

Footer();

// вызываем функцию Footer(). Получим:

//Сегодня: 08.07.05

//Страницу создал Вася

echo "$user, $today";

// выведет запятую, так как

// эти переменные видны только

// внутри функции

?>

**Пример 3. Область видимости при использовани include (,** **)**

Кроме локальных файлов, с помощью *include* можно включать и внешние файлы, указывая их url-адреса. include() – это специальная языковая конструкция, поэтому при использовании внутри условных блоков ее нужно заключать в фигурные скобки.

<?php

/\* Это неверная запись. Получим ошибку.

Мы же вставляем не одну команду,

а несколько, они только записаны

в другом файле \*/

if ($condition) include("first.php");

else include("second.php");

// А вот так правильно.

if ($condition){ include("first.php"); }

else { include("second.php"); }

?>

**Пример** **Использования include() (,** **)**

При использовании *include* возможны два вида ошибок – ошибка вставки (например, нельзя найти указанный файл, неверно написана сама команда вставки и т.п.) или ошибка исполнения (если ошибка содержится во вставляемом файле). В любом случае при ошибке в команде *include* исполнение скрипта не завершается.

##### require

Этот оператор действует примерно так же, как и #include в C++. Все, что мы говорили о *include*, лишь за некоторыми исключениями, справедливо и для *require*. *Require* также позволяет включать в программу и исполнять какой-либо файл. Основное отличие *require* и *include* заключается в том, как они реагируют на возникновение ошибки. Как уже говорилось,*include* выдает предупреждение, и работа скрипта продолжается. Ошибка в *require* вызывает фатальную ошибку работы скрипта и прекращает его выполнение.

*Условные операторы* на require() не влияют. Хотя, если строка, в которой появляется этот оператор, не исполняется, то ни одна строка кода из вставляемого файла тоже не исполняется. Циклы также не влияют на require(). Хотя код, содержащийся во вставляемом файле, является объектом цикла, но вставка сама по себе происходит только однажды.

В реализациях PHP до версии 4.0.2 использование require() означало, что интерпретатор обязательно попытается прочесть вставляемый файл.

#### Решение задачи

Мы хотим создать программу, которую можно было бы использовать для отправки писем (или просто для их генерации) с приглашениями на различные мероприятия множеству пользователей. В предыдущей лекции уже рассматривался подобный случай. Сейчас мы вынесем всю информацию о людях и событиях в отдельный файл data.php и напишем программу, не зависящую от этой информации и ее структуры. В этом случае для того, чтобы, например, расширить список адресатов, не нужно будет изменять скрипт, генерирующий приглашения. Кроме того, можно будет использовать информацию о людях и событиях в других скриптах.

<?php

define("SIGN","С уважением, Вася");

// пусть наша подпись

// будет константой

// информация о событиях

$events = array(

"f" => "день открытых дверей",

"o" => "открытие выставки",

"p" => "бал выпускников");

// имеющаяся информация о людях

// (имя и электронный адрес)

$people = array(

"ivan" => array(

"name" => "Иван Иванович",

"email"=>"user\_ivan@"),

"pit" => array(

"name" => "Петр Петрович",

"email" => "user\_petr@"),

"semen" => array(

"name" => "Семен Семенович"));

// кто куда приглашается

$who\_where["ivan"] = "o" ;

// Иван – на выставку

$who\_where["pit"] = "p";

// Петр – на бал

?>

**Листинг**  **data.php (,** **)**

<?php

require("data.php");

// включаем файл с данными о событиях

foreach($people as $key => $man\_info){

// для каждого человека делаем следующее:

$event\_key = $who\_where[$key];

// получаем событие,

// на которое он приглашается

if ($event\_key<>""){

foreach($man\_info as $key1 => $info){

// получаем имя и email

// конкретного человека

if ($key1=="name")

$str = "Уважаемый (ая), $info";

if ($key1=="email") $email = $info;

}

// составляем приглашение

$str .= "<br>Приглашаем Вас на ".

$events[$event\_key];

switch ($event\_key){

// в зависимости от события

// добавляем какую-нибудь строчку

case "f":

$str .= "<br>Подтвердите Ваше

участие по телефону!";

break;

case "o":

$str .= "<br>Приходите за 15

минут до открытия!";

break;

case "p":

$str .= "<br>Не забудьте подарок :-)";

break;}

$str .= "<br>" . SIGN . "<hr>";

// добавляем подпись

echo $str; // вводим приглашение на экран

/\* если у вас настроена отправка почты с

помощью PHP, то письмо можно отправить

командой mail($email,"Letter",$str); \*/}}?>

Контрольные вопросы:

* 1. Что такое include(), и для чего он нужен?
  2. Что такое require (), и для чего он нужен?

**Организация и заполнение данными сессий и кук, текстовый формат json**

*Цель*: получить навыки организации взаимодействия клиентской и серверной части web-приложения

**Работа с сессиями PHP**

Многие начинающие не понимают, что когда они видят страницу сайта в своем браузере, PHP скрипт уже давно отработал и забыл о нас.

Поэтому, если мы переходим с одной страницы сайта на другую - php скрипт не может запомнить данные с предыдущей страницы, например, значения переменных.

Однако, такой механизм очень нужен, хотя бы для того, чтобы запоминать выбор пользователя или то, что пользователь был авторизован.

На помощь программистам приходят такие механизмы как **сессии** и **cookie** (куки).

**Что такое сессии в PHP?**

**Сессия** - это механизм PHP, который позволяет хранить данные для конкретного пользователя между запусками скрипта.

Мы можем записывать какую-либо информацию в сессию и считывать ее оттуда в следующем запуске этого или другого скрипта сайта.

С помощью сессии можно реализовать авторизацию пользователей, корзину интернет-магазина и другое.

**Инициализируем сессию**

Чтобы записать что-то в сессию ее сначала нужно *инициализировать* с помощью функции**session\_start()**:

<?php

//Инициализируем сессию:

session\_start();

?>

Обратите внимание на то, что в одном скрипте сессия должна инициализироваться **только один раз**, иначе скрипт выдаст ошибку.

**Пишем в сессии**

После инициализации мы можем записать что-нибудь в сессию.

Сделать это не сложно:

<?php

//Инициализируем сессию:

session\_start();

//Пишем в сессию:

$\_SESSION['test'] **=** 'Тест!';

?>

**Читаем из сессий**

После того, как мы что-то записали в сессию, мы можем это оттуда извлечь:

<?php

//Инициализируем сессию:

session\_start();

//Выведем переменную test из сессии:

echo $\_SESSION['test'];

?>

**Возможные проблемы**

Основная проблема при работе с сессией следующая: нельзя делать никакого вывода в браузер до окончания работы с сессиями.

Что такое вывод в браузер?

Это любой символ до **<?php**, например, текст или тег, даже пробел. Кроме того нельзя делать выводы через **echo**, **var\_dump** и **print\_r**.

**Кодировка** вашего документа обязательно должна быть utf-8 **без BOM**. Если она будет просто utf-8, то перед <?php будет вставлен спец. символ, характерный для данной кодировки и сессии работать не будут.

Если вы делаете какие-то вывод в браузер до работы с сессией, то увидите следующую ошибку:*Warning: Cannot send session cookie - headers already sent*.

Эта ошибка также вызывается другими сообщениями об ошибках, так как эти сообщения - это вывод в браузер до старта сессии. Обратите на это внимание.

**Сделаем счетчик на сессиях**

Чтобы продемонстрировать работу с сессиями давайте реализуем **счетчик** количества обновления страницы пользователем.

При каждом обновлении страницы он будет увеличиваться на единицу, а при закрытии браузера - обнуляться (после закрытия нужно подождать 15-25 минут):

<?php

//Инициализируем сессию:

session\_start();

/\*

Переменная $\_SESSION['counter'] будет нашим счетчиком.

Если скрипт запускается первый раз -

она будет пуста, присвоим ей единицу.

Если не первый раз - тогда прибавим единицу.

\*/

if (**!**isset($\_SESSION['counter'])) {

$\_SESSION['counter'] **=** 1;

} else {

$\_SESSION['counter'] **=** $\_SESSION['counter'] **+** 1;

}

//Выведем значение счетчика:

echo 'Вы обновили эту страницу '.$\_SESSION['counter'].' раз!';

/\*

Обновите страницу несколько раз,

посмотрите на то, как увеличивается значение переменной.

Затем закройте браузер, подождите полчаса и откройте снова -

убедитесь в том, что переменная обнулилась!

\*/

?>

**Удалении переменных из сессии**

Переменную сессии можно удалять с помощью функции **unset**:

<?php

//После выполнения этой команды в $\_SESSION['var'] станет null:

unset($\_SESSION['var']);

?>

**Завершение сессии**

Если же вам нужно удалить все переменные сессии для данного пользователя, то вместо **unset**следует воспользоваться функцией **session\_destroy()** (ее можно вызывать только тогда, когда сессия запущена через session\_start):

<?php

//После выполнения этой команды ВСЕ переменные сессии станут null:

session\_destroy();

?>

Мы рассмотрели принудительное завершение сессии. Однако сессия может завершится сама в двух следующих случаях: при закрытии браузера пользователем или по истечении определенного времени, во время которого пользователь не совершает никаких действий на странице (по умолчанию это время около 15-25 минут).

**Какие задачи решает сессия**

Стандартные задачи, которые решают с помощью сессии, это корзина интернет-магазина, форма для заполнения пользователем, разбитая на несколько страниц сайта (на первой мы спрашиваем имя, на второй email и т.д.), [авторизация пользователей на PHP](http://theory.phphtml.net/books/php/auth/).

*Задачи на сессии*

**Задача.** Сделайте две страницы: **index.php** и **hello.php**. При заходе на **index.php** спросите с помощью формы имя пользователя, запишите его в **сессию**. При заходе на **hello.php** поприветствуйте пользователя фразой **"Привет, %Имя%!"**.

**Решение:**

Страница index.php:

**<**form action**=**"" method**=**"GET"**>**

**<**input type**=**"text" name**=**"username"**>**

**<**input type**=**"submit"**>**

**</**form**>**

<?php

//Если форма была отправлена и имя не пустое:

if (**!**empty($\_REQUEST['username'])) {

session\_start(); //стартуем сессию

$\_SESSION['username'] **=** $\_REQUEST['username']; //пишем в сессию

}

?>

Страница hello.php:

<?php

session\_start(); //стартуем сессию

//Если есть данные в сессии об имени пользователя:

if (**!**empty($\_SESSION['username'])) {

echo $\_SESSION['username']; //выведем имя на экран

}

?>

**Задача.** Спросите у пользователя телефон с помощью формы. Затем сделайте так, чтобы **в другой форме** (поля: имя, фамилия, телефон) при ее открытии поле телефон было автоматически заполнено.

Спрашиваем телефон:

**<**form action**=**"" method**=**"GET"**>**

**<**input type**=**"text" name**=**"phone"**>**

**<**input type**=**"submit"**>**

**</**form**>**

<?php

//Если форма была отправлена и телефон не пустой:

if (**!**empty($\_REQUEST['phone'])) {

session\_start(); //стартуем сессию

$\_SESSION['phone'] **=** $\_REQUEST['phone']; //пишем телефон в сессию

}

?>

Другая форма:

<?php

session\_start(); //стартуем сессию

//Если телефон есть в сессии - запишем его в переменную $phone:

if(**!**empty($\_SESSION['phone']))

$phone **=** $\_SESSION['phone'];

else

$phone **=** '';

?>

**<**form action**=**"" method**=**"GET"**>**

**<**input type**=**"text" name**=**"name"**>**

**<**input type**=**"text" name**=**"surname"**>**

**<!--** Заполним атрибут value переменной $phone: **-->**

**<**input type**=**"text" name**=**"phone" value**=**"<?php echo $phone ?>">

<input type="submit"**>**

**</**form**>**

1. Сделайте две страницы: **index.php** и **test.php**. При заходе на **index.php** спросите с помощью формы **страну** пользователя, запишите ее в **сессию**. При заходе на **test.php** выведите страну пользователя.
2. Запишите в сессию время захода пользователя на сайт. При обновлении страницы выводите**сколько секунд назад** пользователь зашел на сайт.
3. Спросите у пользователя **email** с помощью формы. Затем сделайте так, чтобы **в другой форме**(поля: имя, фамилия, пароль, email) при ее открытии поле email было автоматически заполнено.
4. Сделайте **счетчик обновления страницы** пользователем. Данные храните в сессии. Скрипт должен выводить на экран количество обновлений. При первом заходе на страницу он должен вывести сообщение о том, что вы еще не обновляли страницу.
5. Сделайте две страницы: **index.php** и **form.php**. При заходе на **index.php** спросите с помощью формы город и возраст пользователя. На **form.php** сделайте форму с полями 'Имя', 'Возраст', 'Город'. При заходе на **form.php** сделайте так, чтобы поля **'Возраст'** и **'Город'** уже были заполнены.
6. Добавьте в предыдущую задачу страницу **logout.php**. При заходе на нее **разрушайте сессию**пользователя.
7. Реализуйте **тест** по принципу *'одна страница сайта - одна задача'*. Запомните результаты ответов пользователя **в сессию**.

**Задачи для решения Работа со cookie**

**Задача.** Сделайте две страницы: **index.php** и **hello.php**. При заходе на **index.php** спросите с помощью формы имя пользователя, запишите его в **куки**. При заходе на **hello.php** поприветствуйте пользователя фразой **"Привет, %Имя%!"**.

**Решение:**

Страница index.php:

**<**form action**=**"" method**=**"POST"**>**

**<**input type**=**"text" name**=**"username"**>**

**<**input type**=**"submit"**>**

**</**form**>**

<?php

//Если форма была отправлена и имя не пустое:

if (**isset**($\_POST ['username'])) {

//Пишем имя в куки:

setcookie('username', $\_REQUEST['username'], time()**+**3600, '/'); }?>

Страница hello.php:

<?php

//Если есть данные в куки об имени пользователя:

if (**!**empty($\_COOKIE['username'])) {

echo $\_COOKIE['username']; //выведем имя на экран }?>

**Задача.** Спросите у пользователя телефон с помощью формы. Затем сделайте так, чтобы **в другой форме** (поля: имя, фамилия, телефон) при ее открытии поле телефон было автоматически заполнено.

Спрашиваем телефон:

**<**form action**=**"" method**=**"GET"**>**

**<**input type**=**"text" name**=**"phone"**>**

**<**input type**=**"submit"**>**

**</**form**>**

<?php

//Если форма была отправлена и телефон не пустой:

if (**!**empty($\_REQUEST['phone'])) {

//Пишем телефон в куки:

setcookie('phone', $\_REQUEST['phone'], time()**+**3600, '/'); }

?>

Другая форма:

<?php

//Если телефон есть в куки - запишем его в переменную $phone:

if(**!**empty($\_COOKIE['phone']))

$phone **=** $\_COOKIE['phone'];

else

$phone **=** '';

?>

**<**form action**=**"" method**=**"GET"**>**

**<**input type**=**"text" name**=**"name"**>**

**<**input type**=**"text" name**=**"surname"**>**

**<!--** Заполним атрибут value переменной $phone: **-->**

**<**input type**=**"text" name**=**"phone" value**=**"<?php echo $phone ?>">

<input type="submit"**>**

**</**form**>**

**Задачи для решения Работа со cookie**

1. Сделайте две страницы: **index.php** и **test.php**. При заходе на **index.php** спросите с помощью формы **страну** пользователя, запишите ее **в куки** с именем **country**. При заходе на **test.php** выведите страну пользователя.
2. **Удалите куку** с именем **country**.
3. **Установите куку** с именем **age**. Запишите туда случайное число от **10** до **70** (с помощью [mt\_rand](http://theory.phphtml.net/php/math/mt_rand.html)). Сделайте так, чтобы эта кука установилась на 1 час, на 3 часа, на 1 день, на год, на 10 лет, до конца текущего дня, до конца текущего года.

**Задачи**

1. Напишите **оболочку над cookie**. Оболочка должна представлять собой набор функций: сохранение куки, удаление куки, редактирование куки.
2. Сделайте на сайте 5 картинок с товарами. Реализуйте **корзину**. Под каждой картинкой должна быть ссылка 'Положить в корзину'. По нажатию на эту ссылку этот товар должен занестись в корзину (сессия), также должна увеличиться общая сумма, которую должен заплатить пользователь (сумма также должна быть указана под картинками с товарами).
3. Сделайте **счетчик посещения сайта** посетителем. Каждый раз, заходя на сайт, он должен видеть надпись: **'Вы посетили наш сайт % раз!'**.
4. Покажите пользователю **баннер** с кнопкой 'Не показывать больше!'. Если он нажмет на эту кнопку - не показывайте ему баннер в течении месяца.
5. Запомните **дату последнего посещения** сайта пользователем. При заходе на сайт напишите ему, сколько дней он не был на вашем сайте.
6. Спросите **дату рождения** пользователя. При следующем заходе на сайт напишите сколько дней осталось до его дня рождения. Если сегодня день рождения пользователя - поздравьте его!
7. Реализуйте **выбор дизайна** сайта пользователем. Сделайте несколько дизайнов сайта. Пользователь может выбрать один из дизайнов с помощью выпадающего списка. Этот выбор будет сохранен в куки и пользователь, заходя на сайт, всегда будет видеть один и тот же дизайн.

*Контрольные вопросы*

1. Почему контроль заполнения и корректности введенных данных для элементов форм желательно осуществлять и на клиентской, и на серверной стороне?
2. Где подобная проверка более критична с точки зрения безопасности работы Web-приложения?
3. Как серверная сторона Web-приложения может получить значения, заданные в элементах формы перед ее отправкой?
4. Что такое сессионность работы Web-приложения?
5. Для чего она может понадобиться?
6. Какие средства обеспечения сессионности работы Web-приложения?

**Организация подключения и создание запроса к БД**

*Цель:* научиться создавать запросы простые и сложные к готовой базе данных.

***Оформление операций работы с базой данных в виде пользовательских функций, выбираемых при помощи переключателя***

Оформить типовые операции с таблицами: вывод данных, удаление записи, вставка новой записи и редактирование записи в виде отдельных функций, созданных самостоятельно. Для этого из фрагментов типовых операций с таблицами базы данных учебного Web-ресурса, разработанных в предыдущих работах (практические работы 1 — 5), создать новый скрипт в файле **www/admin/index.php**, позволяющий управлять записями книг в базе данных: выводом данных, удалением записи, вставкой новой записи и редактированием записи.

На рис. 6.1 приведена структура таблицы BOOKS учебной базы данных Lib, на основе которой будут демонстрироваться примеры.

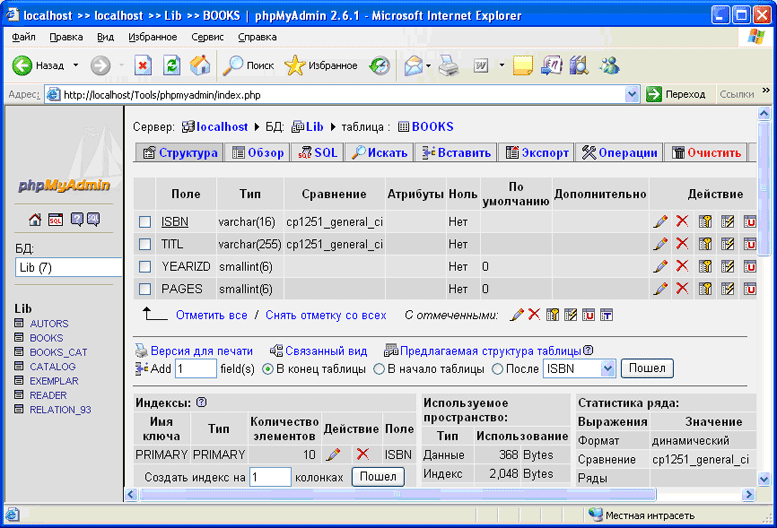


Рис. 6.1. Структура таблицы BOOKS учебной базы данных Lib

Пользовательская функция имеет следующий синтаксис:

**function Имя\_функции (параметр1, параметр2, ... параметрN)**

**{**

**Блок\_действий;**

**return "значение возвращаемое функцией";**

**}**

Если так и написать в php-программе, то работать ничего не будет по следующим причинам.

Во-первых, **Имя\_функции** и имена параметров функции (**пара­метр1, параметр2 и т. д.**) должны соответствовать правилам наименования в PHP (и кириллицу в них лучше не использовать). Имена функций нечувствительны к регистру.

Во-вторых, параметры функции — это переменные языка, поэтому перед названием каждого из них должен стоять знак $. Никаких многоточий ставить в списке параметров нельзя.

В-третьих, вместо слов **Блок\_действий** в теле функции должен находиться любой правильный PHP-код (не обязательно зависящий от параметров).

И наконец, после ключевого слова **return** должно идти корректное PHP-выражение (что-либо, что имеет значение). Возвращаемое значение может быть любого типа, включая списки и объекты. Когда интерпретатор встречает команду **return** в теле функции, он немедленно прекращает ее исполнение и переходит на ту строку, из которой была вызвана функция.

У функции может и не быть входных параметров, как и возвращаемого значения.

Блок описания функций обычно располагается в начале скрипта. Для выполнения задания рекомендуется создать следующий набор функций.

Функция подключения к базе, например:

**function db($login, $passwd)**

**{**

**$dbh=mysql\_connect ("localhost", $login, $passwd) or die ('I cannot connect to the database because: ' . mysql\_e**

**ror());**

**mysql\_select\_db('Lib');**

**}**

Функция выбора данных из таблицы для администрирования:

**function adm()**

**{ … /\*Формирование таблицы со списком книг и кнопками для управления записями \*/**

**}**

На рис. 6.2 приведен пример PHP-кода функции **admin()**, но не обязательно его переписывать точно таким. Можно воспользоваться созданным ранее вариантом формирования таблицы со списком книг, добавив в него кнопки редактирования и удаления записей. Результат работы этой функции должен быть примерно таким, как показано на рис. 6.3.

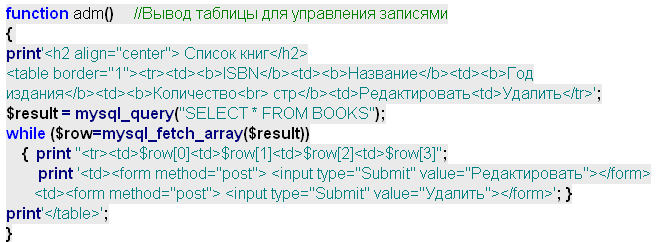


Рис. 6.2. Пример кода функции adm()

Если поле ISBN таблицы BOOKS является автоинкрементным, то его не обязательно выводить в таблицу.

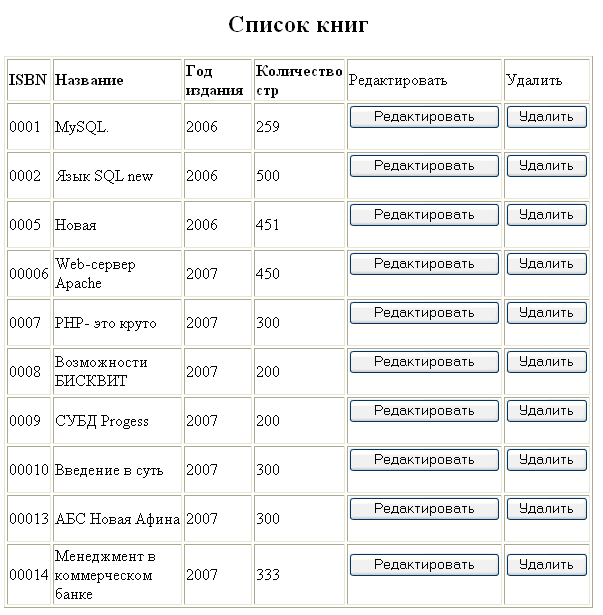


Рис. 6.3. Пример работы функции **adm()**

Проверить действие двух функций можно, вызвав их последовательно по имени. На рис. 6.4 приведен пример полного кода PHP-скрипта, формирующего таблицу рис. 6.3.

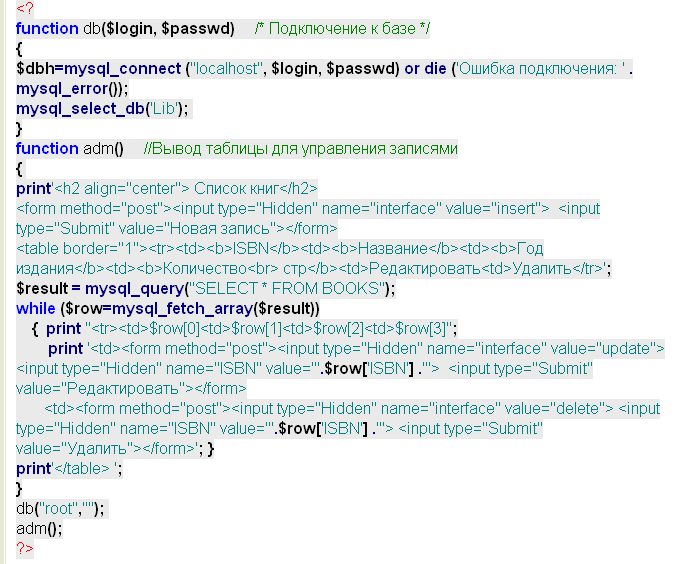


Рис. 6.4. Пример кода PHP-скрипта таблицы со списком книг

После блока из описания двух функций следует PHP-программа с исполняемыми операторами, в том числе с вызовом функций. Например, вызов функции подключения к базе данных будет выглядеть так:

**db("root","");**

Функция **db()** требует в качестве аргументов значения для своих входных параметров **$login** и **$passwd**. На данном этапе при вызове функции **db()** их можно задать в явном виде — имя пользователя СУБД **root** с пустым паролем. Так как эти значения текстовые, то они берутся в кавычки.

Следующим оператором является вызов функции **admin()**, которая не имеет входных параметров.

Следует обратить внимание, что функция **admin()** на рис. 6.4 отличается от варианта, приведенного на рис. 6.2:

во-первых, в формах, вызывающих редактирование и удаление записей (которые в окне браузера формируют соответствующие кнопки) добавлено скрытое поле с именем **interface** (это имя может быть любым) для дальнейшей идентификации вызываемых функций (соответственно редактирования и удаления), а также добавлено еще одно скрытое поле с ключом записи — значением ISBN;

во-вторых, после заголовка перед таблицей расположена форма, отображающая на Web-странице кнопку (Submit), при помощи которой можно организовать запуск функции ввода новой записи в таблицу BOOKS (рис. 6.5).

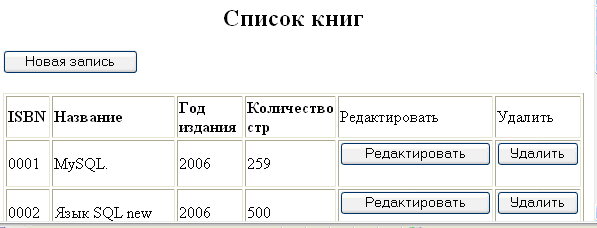


Рис. 6.5. Web-страница списка книг с кнопкой добавления новой записи

Таким образом, из модифицированной функции **admin()** можно организовать запуск одного из трех процессов: добавление новой записи, редактирование любой существующей записи и удаление любой записи. Исполнение этих процессов необходимо также оформить в виде отдельных функций.

Как видно из рис. 6.4, для того, чтобы PHP-скрипт распознал, какую именно функцию необходимо вызвать после нажатия той или иной кнопки на Web-странице (рис. 6.5), рекомендуется ввести идентификатор вызываемой функции, передаваемый в каждой форме при помощи скрытого элемента, например, с именем **interface**:

**<input type="hidden" name="interface" value="insert">.**

Принимается значение, передаваемое через атрибут **value**, аналогично значениям других элементов формы, то есть через элемент массива **$\_POST["interface"]**. Для анализа его значения и вызова соответствующей функции следует использовать переключатель switch, например как показано на рис. 6.6.

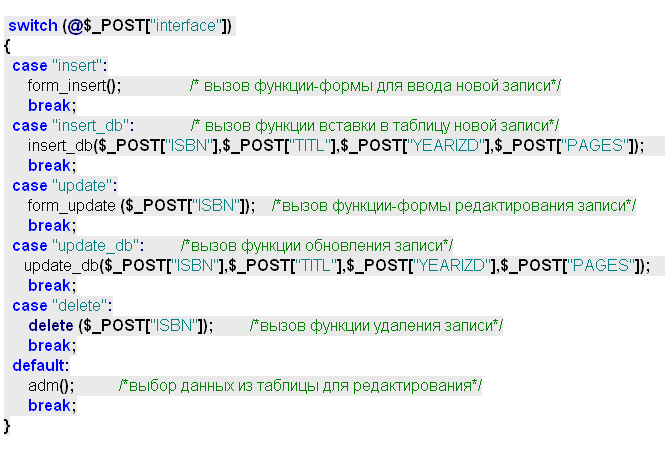


Рис. 6.6. Пример PHP-кода переключателя функций по управлению записями книг

Таким образом, для реализации указанных действий с записями таблицы BOOKS (вставка новой записи, редактирование существующей записи и удаление выбранной записи) необходимо сформировать еще пять функций:

* функцию вывода формы для новой записи;
* функцию вставки новой записи в таблицу;
* функцию вывода формы для редактирования записи;
* функцию обновления записи;
* функцию удаления записи из таблицы.

Еще раз напомним, что каждая функция может иметь входные параметры, список которых в этом случае приводится после имени функции в круглых скобках через запятую (см. рис. 6.6).

Следует также напомнить, что при исполнении скрипта PHP интерпретатор лишь просматривает код функций на предмет их корректности. Функции исполняются лишь при обращении к ним по имени. Количество аргументов вызываемой функции должно совпадать с количеством параметров функции.

Начиная с PHP 4 имеется поддержка списков аргументов переменного размера/variable-length в пользовательских функциях. Это довольно легко делается функциями [**func\_num\_args()**](mk:@MSITStore:F:\РЭП\php42.chm::/f/func-num-args.html)(возвращает количество аргументов, переданных функции), [**func\_get\_arg()**](mk:@MSITStore:F:\РЭП\php42.chm::/f/func-get-arg.html) (возвращает элемент из списка аргументов), [**func\_get\_args()**](mk:@MSITStore:F:\РЭП\php42.chm::/f/func-get-args.html)(возвращает массив — список аргументов функции). Специального синтаксиса не нужно, и списки аргументов могут по-прежнему явно предоставляться в определениях функций и будут вести себя как обычно. Однако для обращения к полям таблицы в SQL-запросе количество параметров не может быть переменным.

Функция вывода формы для новой записи не имеет параметров и должна выводить форму типа рис. 6.7.



Рис. 6.7. Пример формы ввода новой записи

Такая форма ранее уже использовалась. Требуется лишь скопировать ее код в файл **www/admin/index.php** и оформить в виде функции, например с именем **insert()**. Если в таблице BOOKS поле ISBN автоинкрементное, то создавать поле формы для его ввода не требуется. Однако в эту форму необходимо добавить скрытое поле с идентификатором для запуска функции-обработчика данных, которая формирует и выполняет SQL-запрос к базе данных на вставку новой записи. Пример кода для скрытого поля (применительно к примеру, представленному на рис. 6.6):

**<input type="hidden" name="interface" value="insert\_db">.**

Естественно, что далее требуется создать отдельную функцию, которая будет выполнять такой запрос. Применительно к примеру на рис. 6.6 эта функция может иметь код, представленный на рис. 6.8.

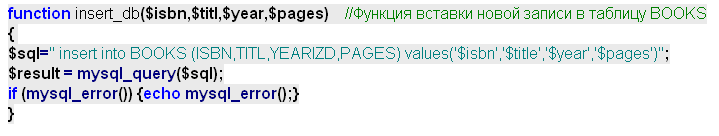


Рис. 6.8. Пример PHP-кода функции вставки новой записи в таблицу BOOKS

Такой SQL-запрос также раньше уже отрабатывался, поэтому можно использовать его код для создания данной функции. Следует обратить внимание, что входные параметры функции, являясь внутренними переменными функции, получают свои конкретные значения при вызове функции из переключателя **switch** (см. рис. 6.6).

Для автоматического возврата к странице редактирования списка книг в функциях (например, после выполнения удаления записи или вставки новой записи) можно воспользоваться перенаправлением, реализуемым средствами HTML:

**print '<HTML><HEAD>**

**<META HTTP-EQUIV="Refresh" CONTENT="0; URL=index.php">**

**</HEAD></HTML>';**

В приведенном примере перенаправление будет выполнено на адрес **index.php** с задержкой **0 с**. При таком обращении к файлу index.php отсутствуют элементы массива $\_POST, поэтому переключатель **switch** передаст управление ветви **default** с вызовом функции **adm()**.

Код перенаправления целесообразно исполнять, если нет ошибки выполнения SQL-запроса, то есть в операторе **if**:

**if (mysql\_error()) {echo mysql\_error();}**

**else { код перенаправления }**

Аналогично процессу вставки новой записи реализуется и процесс редактирования выбранной записи (тоже при помощи двух функций — функции вывода формы и функции-обработчика), который имеет следующие отличия:

* перед выводом формы редактирования должен формироваться запрос к базе по ключу записи (полю ISBN), по результату которого в элементы формы вносятся имеющиеся значения (при помощи атрибутов **value**);
* ключ не нужно подвергать редактированию, а передавать его из этой формы функции-обработчику (функции обновления записи) можно при помощи скрытого поля.

На рис. 6.9. приведен пример внешнего вида формы редактирования данных книги, а на рис. 6.10 — пример PHP-кода функции, реализующей вывод такой формы по заданному значениею ISBN.

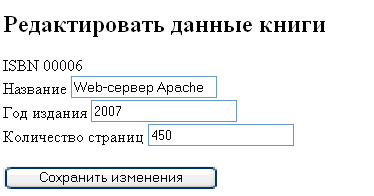


Рис. 6.9. Пример формы для редактирования записи

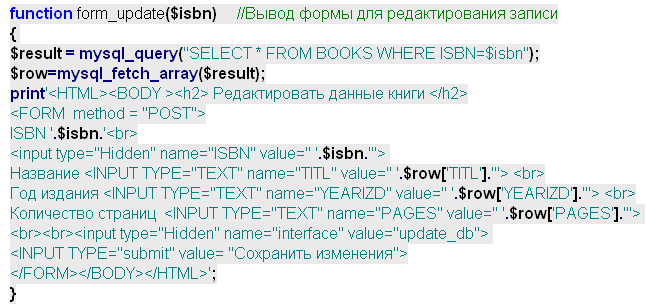


Рис. 6.10. Пример PHP-кода функции вывода формы для редактирования данных книги

Функция обновления записи осуществляет SQL-запрос к базе по обновлению записи в таблице BOOKS, определяемой ключом ISBN.

Функция удаления записи в качестве входного параметра принимает только ключ записи — значение поля ISBN и выполняет соответствующий SQL-запрос. На рис. 6.11 приведен пример PHP-кода функции обновления записи.

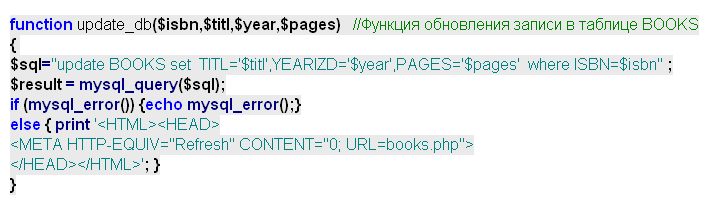


Рис. 6.11. Пример PHP-кода функции обновления записи в таблице BOOKS

Процедура удаления записи самая простая и реализуется при помощи одной функции, принимающей в качестве входного параметра ключ записи — значение ISBN. На рис. 6.12 представлен пример PHP-кода функции удаления записи из таблицы BOOKS. В этом примере реализовано также перенаправление на файл index.php после успешного выполнения запроса.

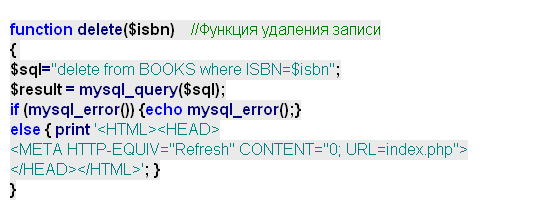


Рис. 6.12. Пример PHP-кода функции удаления записи из таблицы BOOKS

***6.2. Создание подключаемого конфигурационного файла с входными параметрами PHP скрипта***

В PHP-скриптах рекомендуется все конкретные значения, связанные с хостингом и файловой структурой Web-сервера, задавать в отдельном конфигурационном файле, подключаемом ко всем скриптам при помощи функции **include()** или **require()**. Такая технология позволяет легко вносить изменения в настройки скриптов при смене хостинга.

Следует помнить, что когда в скрипт подключается другой файл, разбор переходит из режима PHP в режим HTML в начале целевого файла и вновь продолжатся после конца. Исходя из этого любой код внутри файла назначения, который должен выполняться как PHP-код, обязан быть заключен в правильные стартовый и конечный тэги РНР. С другой стороны, подключаемый файл может быть чисто файлом с HTML-кодом, хотя и будет вызываться из PHP-скрипта. Это особенно полезно при использовании в PHP-скриптах сценариев на языке JavaScript.

Создать файл **www/admin/config.php** и собрать в него переменные с входными данными, например данными работы с базой:

**$bd = 'db';**

**$login = 'root';**

**$passwd = "";**

И т.п.

Затем подключить этот конфигурационный файл в начале создаваемого скрипта при помощи функции **include**(**'config.php'**).

***6.3. Оптимизация кода скрипта***

С целью оптимизации кода скрипта поместите все текстовые строки в одинарные кавычки с присоединением к ним переменных, если есть в этом необходимость. Двойные кавычки используйте лишь при множественном использовании в строке различных переменных.

Устраните многократные повторения операторов **print** или **echo** в строчках, следующих друг за другом.

Проверьте, чтобы не было обращений к базе данных внутри цикла. Если такие случаи есть, то преобразуйте функцию так, чтобы обращение к базе было вне цикла.

***Задание на самостоятельную работу***

Оформить остальные операции по управлению данными учебной базы данных «Библиотека» при помощи пользовательских функций с соблюдением правил оптимизации PHP-кода.

1. Дана таблица с работниками. Реализуйте ее вывод на экран в следующем виде:

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **name** | **age** | **salary** |
| 1 | Дима | 23 | 400 |
| 2 | Петя | 25 | 500 |
| 3 | Вася | 23 | 500 |
| 4 | Коля | 30 | 1000 |
| 5 | Иван | 27 | 500 |
| 6 | Кирилл | 28 | 1000 |

1. Сделайте в таблице всех работников ссылку **'удалить'**. По нажатию на эту ссылку из БД должна **удаляться** запись с этим **id** (его следует передавать через GET-параметр del\_id).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **name** | **age** | **salary** | **удаление** |
| 1 | Дима | 23 | 400 | [удалить](http://code.mu/tasks/php/practice/praktika-php-dlya-novichkov.html) |
| 2 | Петя | 25 | 500 | [удалить](http://code.mu/tasks/php/practice/praktika-php-dlya-novichkov.html) |
| 3 | Вася | 23 | 500 | [удалить](http://code.mu/tasks/php/practice/praktika-php-dlya-novichkov.html) |
| 4 | Коля | 30 | 1000 | [удалить](http://code.mu/tasks/php/practice/praktika-php-dlya-novichkov.html) |
| 5 | Иван | 27 | 500 | [удалить](http://code.mu/tasks/php/practice/praktika-php-dlya-novichkov.html) |
| 6 | Кирилл | 28 | 1000 | [удалить](http://code.mu/tasks/php/practice/praktika-php-dlya-novichkov.html) |

1. Сделайте форму добавления нового работника.
2. Сделайте колонку 'редактировать' для каждого работника. Там должна быть ссылка, по переходу на которую появится страница с формой редактирования работника.
3. Над таблицей с работниками сделайте инпут, в который вводится зарплата. По нажатию на кнопку следует вывести таблицу работников с введенной зарплатой.
4. Над таблицей с работниками сделайте инпут, в который вводится зарплата. По нажатию на кнопку следует вывести таблицу работников с введенной зарплатой.
5. Сделайте колонку 'удалить', в которой для каждого работника будет стоять чекбокс. Под таблицей сделайте кнопку, по нажатию на которую будут удалены те работники, для которых чекбокс был отмечен.

**Извлечение данных из БД**

*Цель:* научиться извлекать данные из базы данных в таблицу, на страницу, и между страницами на сервере.

**Упражнение 1: Вывод данных из базы на страницу**

* этом упражнении на главную страницу сайта необходимо вывести все заметки из таблицы БД **notes**.
  1. Переименуйте **blog.html** в **blog.php** (старую html-страницу после переименования можно удалить).
  2. Удалите записи на странице. Должны остаться только приветствие и навигационное меню.
  3. Создайте соединение с сервером. Оно у нас уже реализовано в файле **MySitDB.php** –файл надо просто включить с помощью функции ***require\_once(),*** в качестве параметра передав ей путь к файлу

(«**Connections/MySiteBD.php**»):

*<?php require\_once ("connections/MySiteDB.php"); ?>*

1. Далее необходимо вывести записи (строки) на страницу сайта из таблицы notes. Сначала надо реализовать запрос на выборку. Для этого:

4.1. выберите БД;

4.2. создайте SQL-запрос на выборку данных из таблицы (SELECT *fields* FROM *tableName*).ЗдесьSELECT –оператор выбора полей,FROM – оператор выбора таблицы-источника полей.

* + *Если вам необходимо выбрать все поля таблицы (как в данном случае), то запрос можно построить так: SELECT \* FROM tablename, где символ «\*» обозначает все поля таблицы.*

4.3. Реализуйте запрос на выборку.

1. Далее необходимо вывести запись на страницу сайта. Для этого используется функция ***mysqli\_fetch\_array()***. Параметром функции является переменная, содержащая результат выполнения запроса к БД (в данном случае – реализации запроса на выборку); собственно функция получает по одной записи из таблицы за один раз. Каждая запись возвращается в виде массива.
2. Для вывода информации из массива по отдельным элементам необходимо придерживаться следующего синтаксиса:

*//Вывод элементов массива*

*echo $note ['id'], "<br>";*

*echo $note ['created'], "<br>";*

*echo $note ['title'], "<br>";*

*echo $note ['article'], "<br>";*

Сейчас из таблицы с помощью функции ***mysqli\_fetch\_array()*** выводится только одна запись. С помощью цикла необходимо сделать так, чтобы выводились все записи из таблицы. Для этого необходимо изменить часть кода следующим образом:

*//Использование цикла while*

*while ($note = mysqli\_fetch\_array($select\_note)){ echo $note ['id'], "<br>";*

*echo $note ['created'], "<br>";*

*echo $note ['title'], "<br>";*

*echo $note ['article'], "<br>";}*

Здесь переменной с именем *$select\_note* присваевается результат выполнения запроса к БД *mysqli\_query()*.

**Упражнение 2: Обмен данными между серверными страницами**

Каждая заметка на главной странице блога может быть прокомментирована. Для реализации этой функции необходимо сделать из заголовка каждой заметки гиперссылку, перейдя по которой посетитель попадет на страницу со списком комментариев к выбранной заметке. Кроме того, на этой же странице должна отображаться сама выбранная для комментирования заметка.

Следовательно, необходимо реализовать механизм обмена данными между страницами таким образом, чтобы при переходе по гиперссылке передавалась информация о том, какая именно заметка была выбрана.

Для этого необходимо ввести некий идентификатор , значение которого будет совпадать с id комментируемой заметки, и который будет передаваться при переходе по гиперссылке.

1. Создание гиперссылки

1.1. Создайте новую страницу comments.php, которая будет содержать комментарии к выбранной заметке.

1.2. Реализуйте соединение с сервером.

1.3. Реализуйте подключение к БД.

1.4. Для передачи идентификатора заметки введем аргумент **note**. В качестве значения он будет получать значение поля ***id*** таблицы **notes**.

1.5. На странице **blog.php** найдите фрагмент кода, передающего заголовок заметки ***title (echo $ note [‘title’];)***. Его необходимо отредактировать таким образом, чтобы он стал гиперссылкой на страницу комментариев **comments.php**, а также передавал ***id*** выбранной заметки:

*while ($note = mysqli\_fetch\_array($select\_note)){ echo $note['id'], "<br>";*

*?>*

*<a href="comments.php?note=<?php echo $note['id']; ?>"> <?php echo $note ['title'], "<br>";?></a>*

*<?php*

*echo $note ['created'], "<br>";*

*echo $note ['content'], "<br>";?>*

Здесь мы создаем гиперссылку на страницу **comments.php** и в этой гиперссылке передаем идентификатор **note**, значение которого равно значению элемента массива ***$note[‘id’],*** т.е. значению ***id*** заметки.

1. Страница **comments.php**

2.1. Перейдите на страницу **comments.php**. На данной странице должны отображаться комментарии к выбранной записи, а также сама комментируемая запись (для удобства посетителя сайта).

2.2. Данную задачу можно выполнить по аналогии с выводом заметок на странице **blog.php**. Основное отличие заключается в том, что вначале необходимо со станицы **blog.php** получить переданный с помощью идентификатора **note** id заметки. Это делается с помощью метода $\_GET:

*//Переменной $note\_id необходимо присвоить id заметки, переданной*

* *помощью метода $\_GET со страницы blog.php $note\_id = $\_GET['note'];*

2.3. Далее необходимо вывести значения полей created, title, content из таблицы **notes** для заметки с полученным id. Для этого используется SQL запрос

SELECT… FROM… WHERE…

* нем с помощью оператора ***SELECT*** выбираем необходимые поля таблицы; с помощью ***FROM*** определяем таблицу-источник выборки; ***WHERE*** задает условие отбора, по которому выбираем заметку с выбранным ***id***:

*//Формируем SQL-запрос на выборку с учетом переданного id заметки*

*$query = "SELECT created, title, article FROM notes WHERE id = $note\_id";*

2.4. После формирования SQL-запроса его необходимо реализовать с помощью функции ***mysqli\_query()*** и вывести данные на страницу с помощью функции ***mysqli\_fetch\_array().***

2.5. Затем аналогичным образом выведите комментарии к выбранной заметке. Обратите внимание, что SQL-запрос на выборку комментариев должен строиться следующим образом:

*$query\_comments = "SELECT \* FROM comments WHERE art\_id = $note\_id";*

* + условии WHERE мы реализуем поддержку связи между таблицами, которые связаны по полям ***id*** (таблица **notes**) и ***art\_id*** (таблица **comments**).
  + переменной **$note\_id** содержится ***id*** выбранной заметки, следовательно, для выбора комментариев к этой заметке необходимо, чтобы значение поля ***art\_id***created также было равно **$note\_id.**

1. Проверьте корректность данных между страницами **blog.php** и **comments.php**.При переходе по ссылке с **blog.php** на **comments.php** в

адресной строке браузера должен отображаться ***id*** выбранной заметки, переданный с помощью идентификатора **note**.

1. Для того, чтобы выводились все комментарии, а не только первый – реализуйте цикл.
2. Если у заметки нет ни одного комментария – об этом надо сообщить.

5.1. Под областью комментариев добавьте надпись «*Эту запись еще* *никто не комментировал*».

5.2. В коде программы создайте циклы с условием ***if***: если хотя бы один комментарий существует – он должен быть выведен (т.е. элементы массива должны быть отображены); если количество комментариев равно нулю – должна выводиться надпись «*Эту* *запись еще никто не комментировал*».

**Упражнение 3: Вывод данных из базы в таблицу**

<table>

<?php $result = mysql\_query("SELECT id, name FROM mytable");  
while ($row = mysql\_fetch\_array($result)) {?>  
 <tr>  <td> <?php echo $row["id"] ?> </td>

<td> <?php echo $row["name"] ?> </td>

…… </tr>

<?php } ?>  
</table>

<table>

<?php $result = mysql\_query("SELECT id, name FROM mytable");  
while ($row = mysql\_fetch\_array($result)) {?>  
 <tr>

<td> <?php echo $row["id"] ?> </td>

<td> <?php echo $row["name"] ?> </td>

…… </tr>

<?php } ?>  
</table>

<table>

<?php $result = mysql\_query("SELECT id, name FROM mytable");  
while ($row = mysql\_fetch\_array($result)) {?>  
 <tr>

<td> <?php echo $row["id"] ?> </td>

<td> <?php echo $row["name"] ?> </td>

…… </tr>

<?php } ?>  
</table>

<table>

<?php $result = mysql\_query("SELECT id, name FROM mytable");  
while ($row = mysql\_fetch\_array($result)) {?>  
 <tr>

<td> <?php echo $row["id"] ?> </td>

<td> <?php echo $row["name"] ?> </td>

……

 </tr>

<?php } ?>  
</table>

<table>

<?php $result = mysql\_query("SELECT id, name FROM mytable");  
while ($row = mysql\_fetch\_array($result)) {?>  
 <tr>

<td> <?php echo $row["id"] ?> </td>

<td> <?php echo $row["name"] ?> </td>

……

 </tr>

<?php } ?></table>

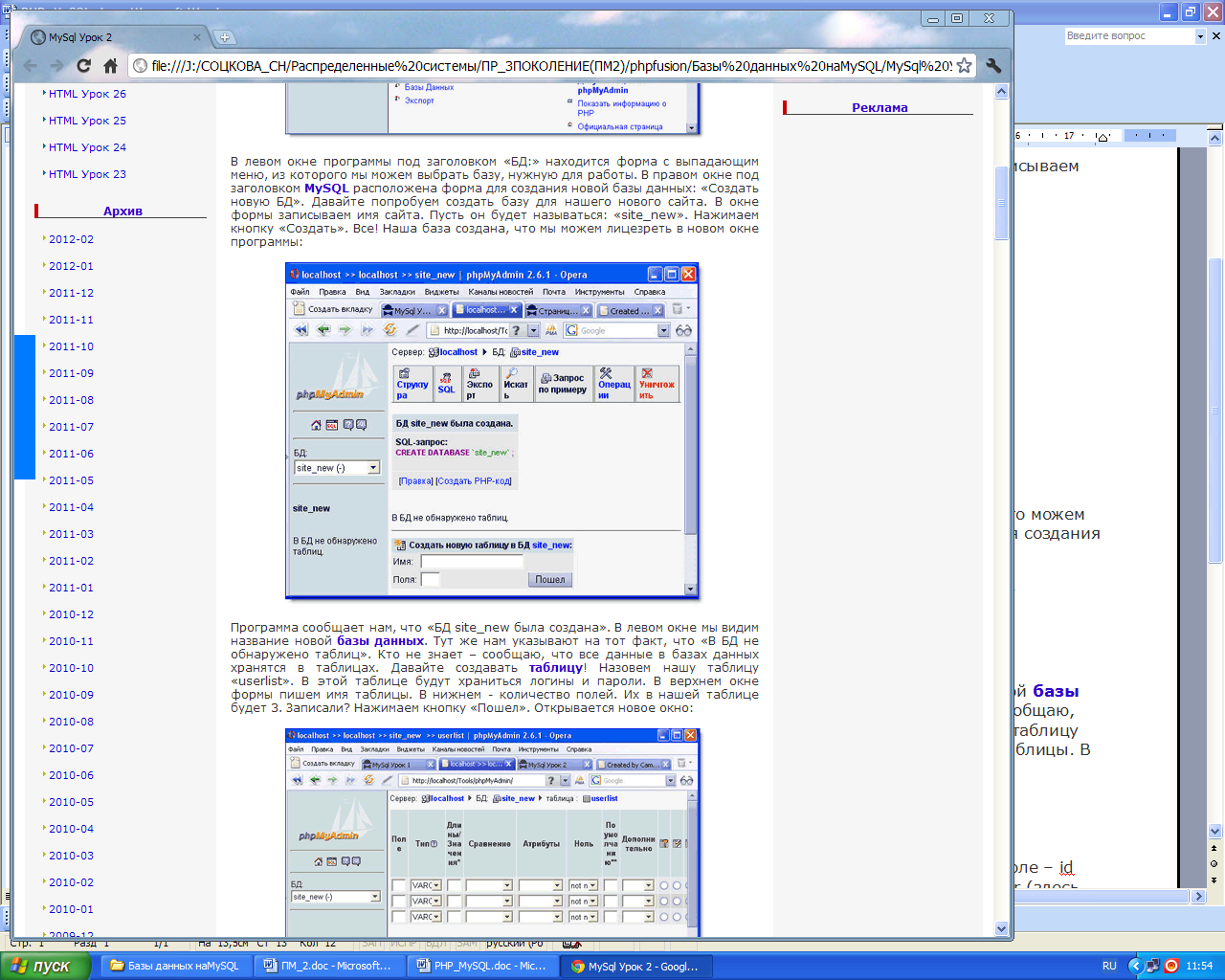
**Добавление данных в БД**

*Цель:* формирование умений применение PHP для доступа к базам данных, и формирование умений работы с MySQL.

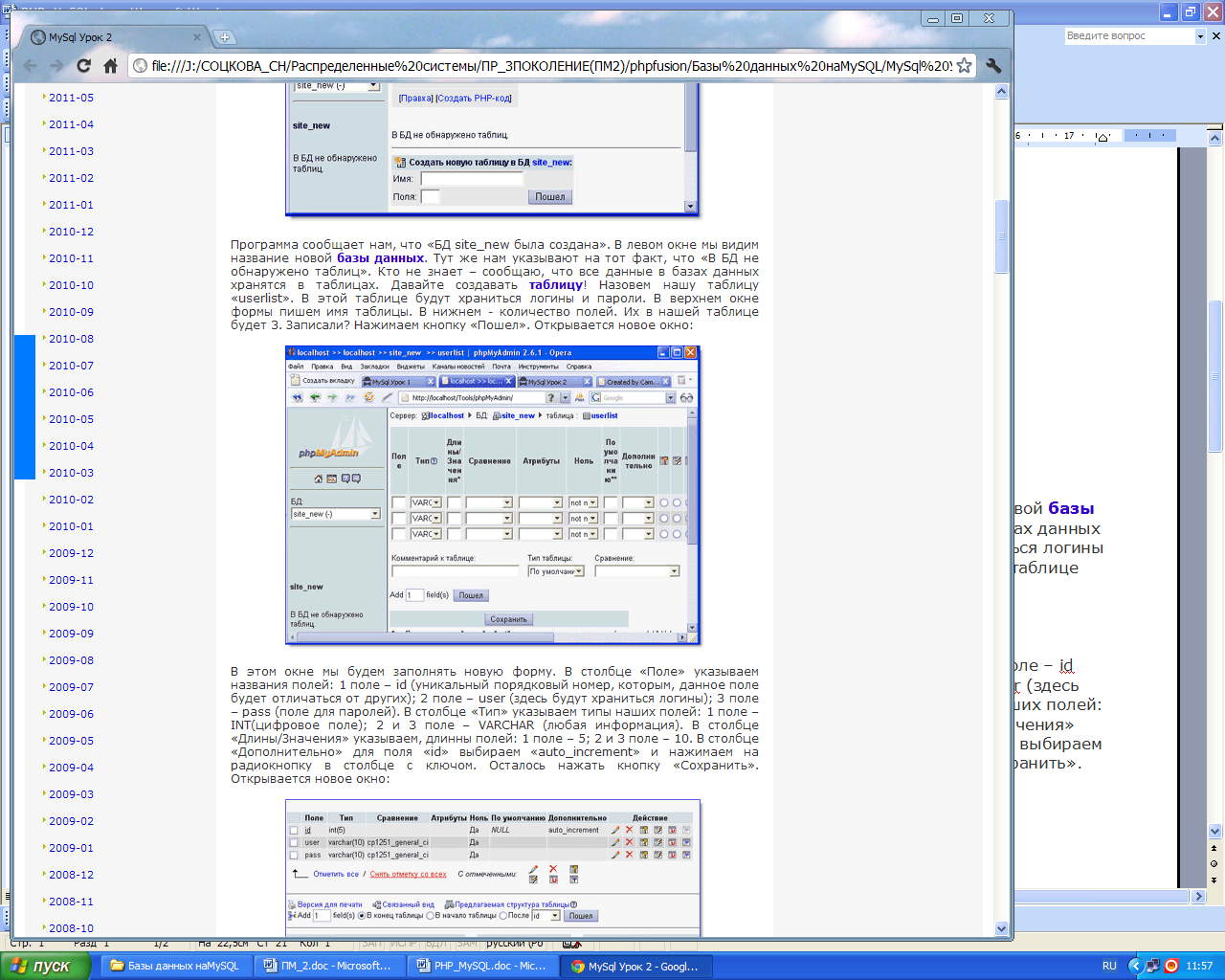
**Задание 1.** Создать БД в PHP.

1 Создайте базу данных ***itrem.***

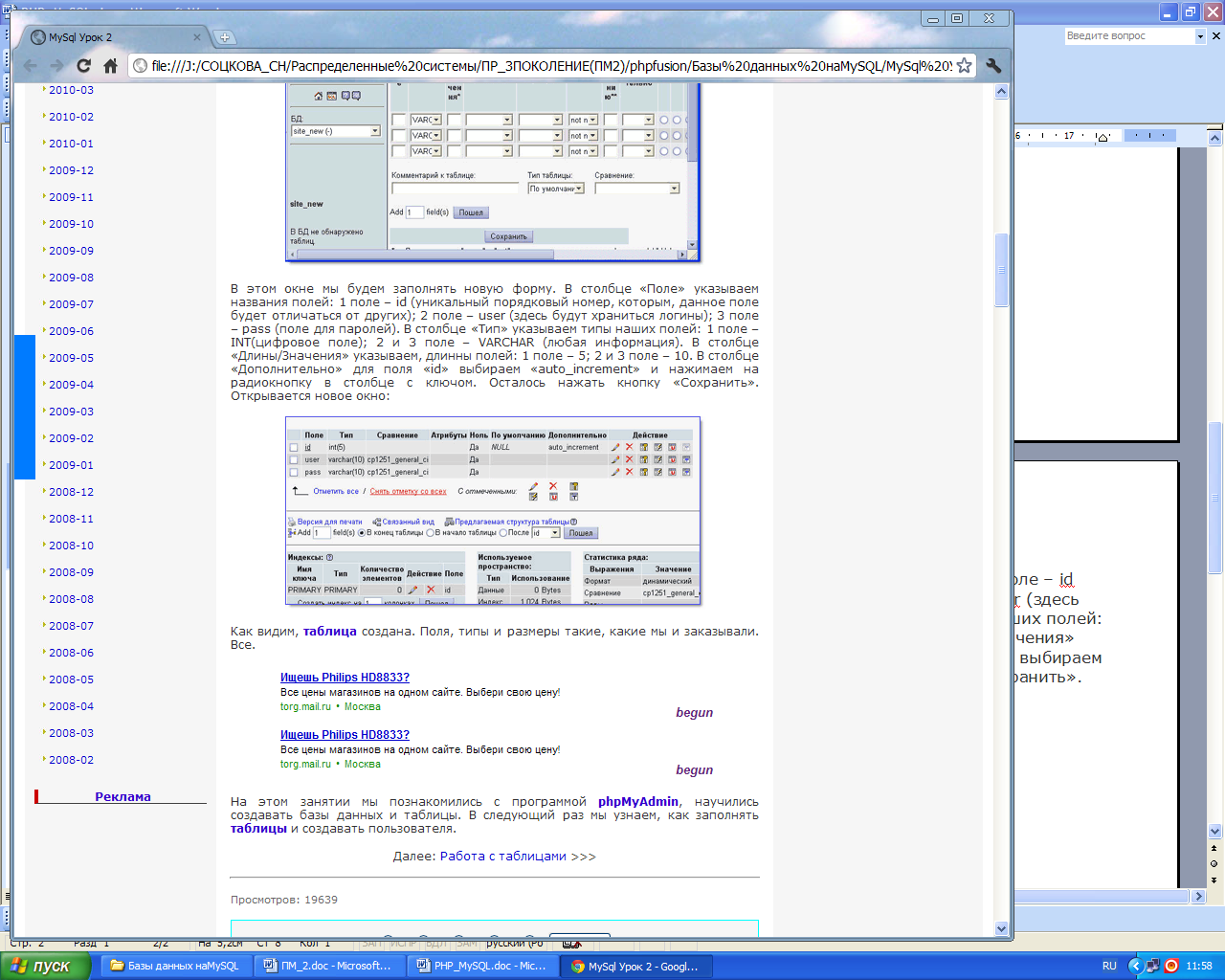
2 Создайте БД «site\_new».



2.1 Программа сообщает нам, что «БД site\_new была создана». В левом окне мы видим название новой **базы данных**. Тут же нам указывают на тот факт, что «В БД не обнаружено таблиц». Все данные в базах данных хранятся в таблицах. Создаем **таблицу.** Назовем таблицу «userlist». В этой таблице будут храниться логины и пароли. В верхнем окне формы пишем имя таблицы. В нижнем - количество полей. Их в нашей таблице будет 3. Нажимаем кнопку «Пошел». Открывается новое окно:

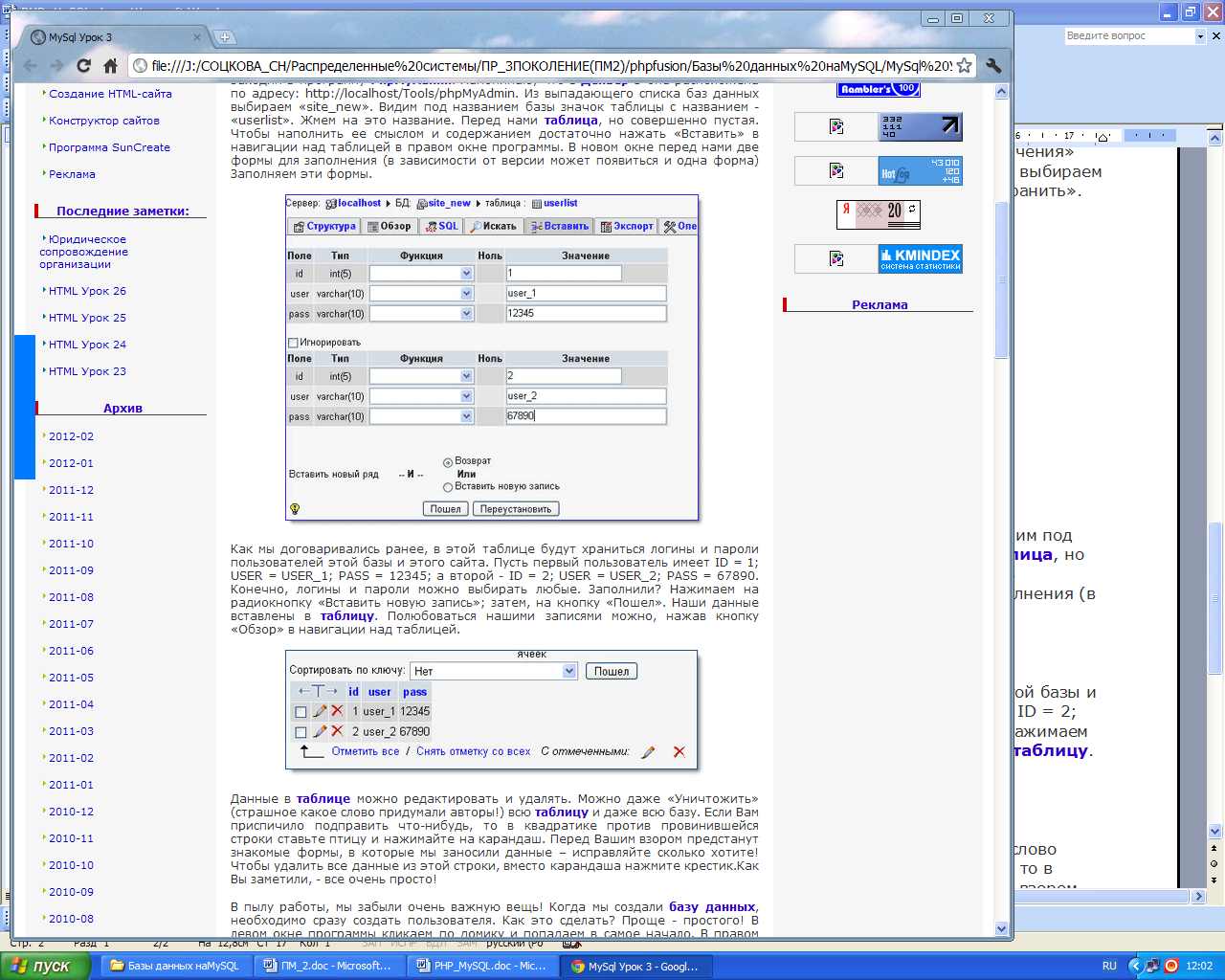


В этом окне мы будем заполнять новую форму. В столбце «Поле» указываем названия полей: 1 поле – id (уникальный порядковый номер, которым, данное поле будет отличаться от других); 2 поле – user (здесь будут храниться логины); 3 поле – pass (поле для паролей). В столбце «Тип» указываем типы наших полей: 1 поле – INT(цифровое поле); 2 и 3 поле – VARCHAR (любая информация). В столбце «Длины/Значения» указываем, длинны полей: 1 поле – 5; 2 и 3 поле – 10. В столбце «Дополнительно» для поля «id» выбираем «auto\_increment» и нажимаем на радиокнопку в столбце с ключом. Осталось нажать кнопку «Сохранить». Открывается новое окно:

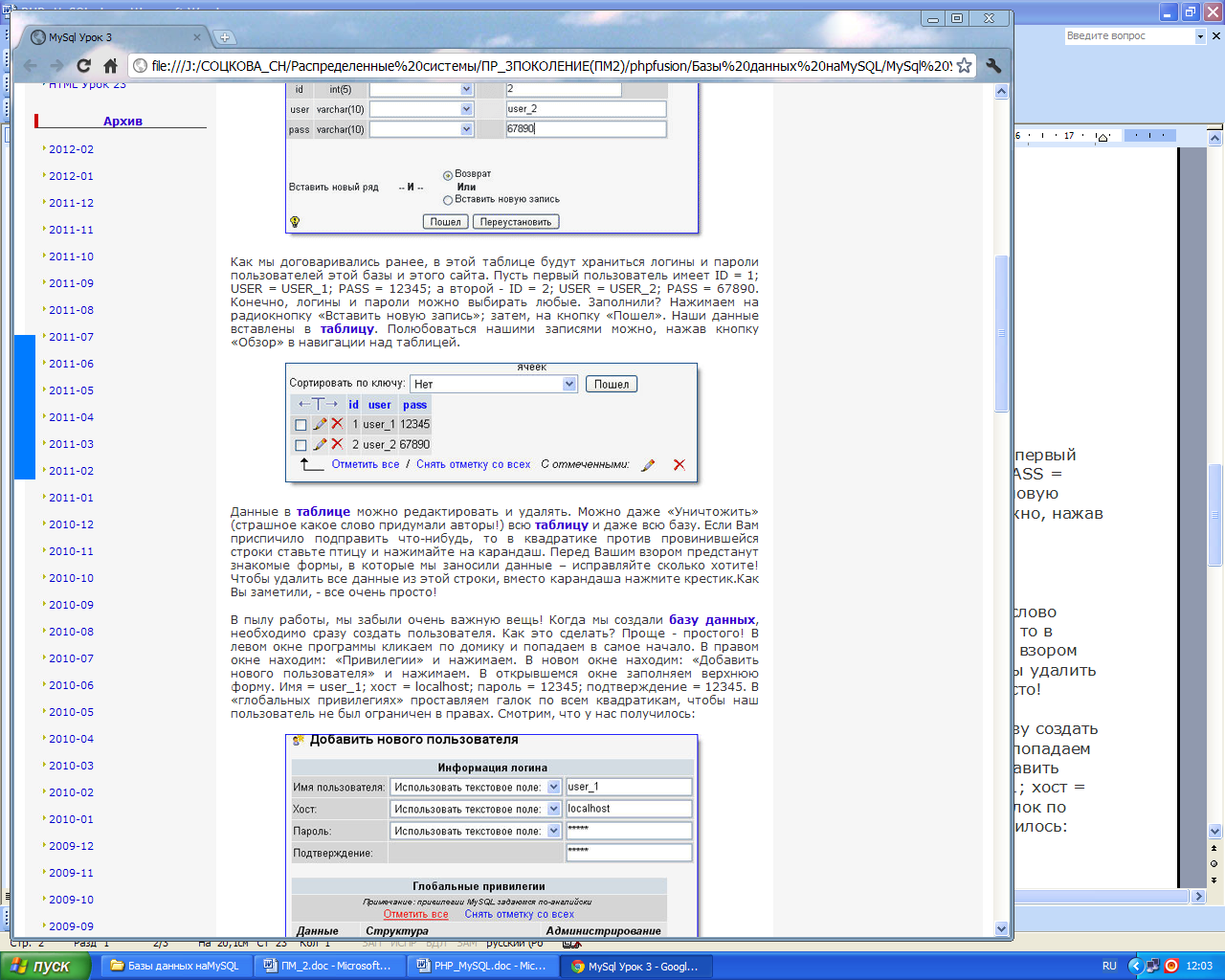


**Таблица** создана. Поля, типы и размеры указаны. Закрываем PHPMyAdmin.

**2.2** Заходим в программу **PhpMyAdmin**. Из выпадающего списка баз данных выбираем «site\_new». Видим под названием базы значок таблицы с названием - «userlist». Жмем на это название. Перед нами **таблица**, но совершенно пустая. Чтобы наполнить ее смыслом и содержанием достаточно нажать «Вставить» в навигации над таблицей в правом окне программы. В новом окне перед нами две формы для заполнения (в зависимости от версии может появиться и одна форма) Заполняем эти формы.

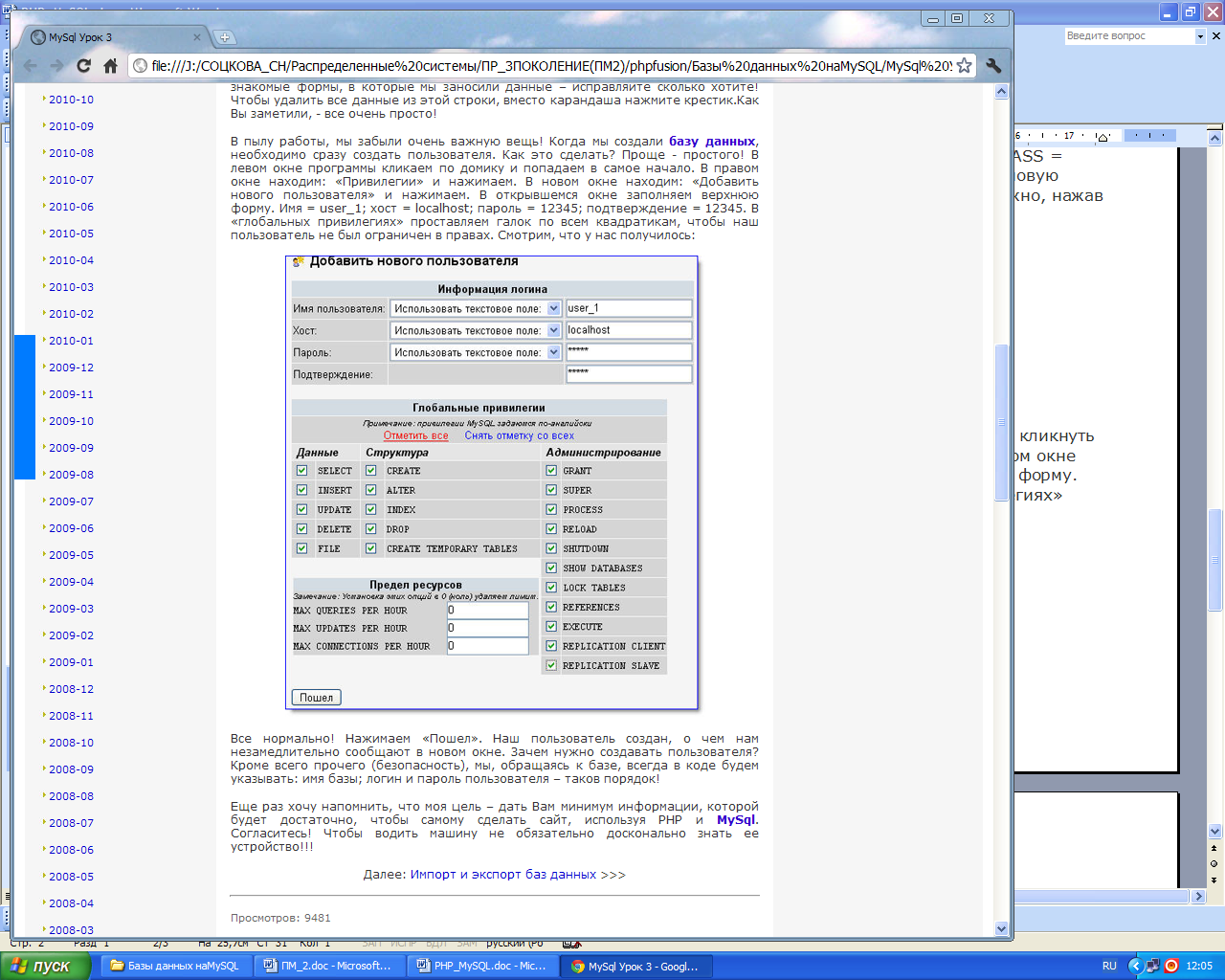


2.3 В этой таблице будут храниться логины и пароли пользователей этой базы и этого сайта. Пусть первый пользователь имеет ID = 1; USER = USER\_1; PASS = 12345; а второй - ID = 2; USER = USER\_2; PASS = 67890. Конечно, логины и пароли можно выбирать любые. Нажимаем на радиокнопку «Вставить новую запись»; затем, на кнопку «Пошел». Наши данные вставлены в **таблицу**. Просмотреть записи можно, нажав кнопку «Обзор» в навигации над таблицей.



Данные в **таблице** можно редактировать и удалять. Чтобы удалить все данные из строки, вместо карандаша нажмите крестик.

2.4 Когда создали **базу данных**, необходимо сразу создать пользователя. В левом окне программы кликнуть по домику и попадаем в самое начало. В правом окне находим: «Привилегии» и нажимаем. В новом окне находим: «Добавить нового пользователя» и нажимаем. В открывшемся окне заполняем верхнюю форму. Имя = user\_1; хост = localhost; пароль = 12345; подтверждение = 12345. В «глобальных привилегиях» проставляем галок по всем квадратикам, чтобы наш пользователь не был ограничен в правах.



Нажимаем «Пошел». Пользователь создан, о чем сообщают в новом окне. Создавать пользователя нужно для осуществления безопасности БД.

Возможность соединиться с БД, выполнить запрос/query, напечатать результирующие ряды и отсоединиться от БД MySQL.

**Создаем управляющий PHP скрипт.**

 Для начала нам необходимо соединиться с базой данных, введя данные созданного нами пользователя и занести получившийся php код в самый верх страницы (сразу под тегом <html>):

*<?php*

*$connect = mysql\_connect("localhost", "root", "") or die("Невозможно установить соединение!".mysql\_error());*

*mysql\_select\_db(site\_new,$connect) or die("Невозможно установить связь с БД!".mysql\_error());*

*//создать запрос*

*$sql = "SELECT \* FROM userlist";*

*$result = mysql\_query($sql) or die(mysql\_error());*

*//Отобразить данные из БД на web-странице в виде таблицы*

*$table = "<table border=2>";*

*$table.="<tr>";*

*$table .= "<td>№</td>";*

*$table .= "<td>Пользователь</td>";*

*$table .= "<td>Пароль</td>";*

*$table .= "</tr>";*

*while (list($id,$user,$pass) = mysql\_fetch\_array($result))*

*{*

*$table .= "<tr>";*

*$table .= "<td>".$id."</td>";*

*$table .= "<td>".$user."</td>";*

*$table .= "<td>".$pass."</td>";*

*$table .= "<td style='height:100px;'></td>";*

*$table .= "</tr>";*

*}*

*$table .= "</table> ";*

*echo $table;*

*?>*

Выносим управляющий код во внешний файл.

Вынесем php код в отдельный файл, с названием db\_connect.php и добавим его на каждую страницу сайта при помощи:

*<?php*

*include ("blocks/db\_connect.php");*

*?>*

**Добавление записей в БД**

Необходимо создать HTML-файл, в котором размещены формы для ввода необходимых записей:

***f\_insert.html***

<html>

<head>

<body>

<h1 align='center'>Введите данные в свободные поля </h1>

<form name="my\_form" action="insert.php" method="post">

<table border='2' align='center'>

<tr>

<td>Введите №:<br /><input type="edit" name="add\_id"><br /></td>

</tr>

<tr>

<td>Введите имя:<br /><input type="edit" name="add\_user"><br /></td>

</tr>

<tr>

<td>Введите пароль:<br /><input type="edit" name="add\_pass"><br /></td>

</tr>

<tr>

<td><input name="Submit" type=submit value="Добавить"><br> </br></td>

</tr>

</form>

</table>

</head>

<html>

Затем создать файл .php, который позволяет произвести добавление записей в БД:

***Insert.php***

<html>

<head>

<meta http-equiv="refresh" content="3; db\_connect.php">

</head>

<?php

$id\_add=$\_POST['add\_id'];

$user\_add=$\_POST['add\_user'];

$pass\_add=$\_POST['add\_pass'];

if (trim($id\_add) and trim($user\_add) and trim($pass\_add))

{

$connect = mysql\_connect("localhost", "user\_1", "12345")

or die("Could not connect".mysql\_error());

echo "Connected successfully";

mysql\_select\_db("site\_new",$connect);

$result = mysql\_query("INSERT INTO userlist(id,user,pass) VALUES('$id\_add', '$user\_add', '$pass\_add')");

if ($result==true)

{

echo "<div align=center> <div class=success>Данные успешно внесены</div> </div>";

}

else

{

echo "<div align=center> <div class=error>Ошибка при добавлении данных</div> </div>";

}

mysql\_close($connect);

}

else

{

echo "<div align=center> <div class=error>Заполните все поля и повторите ввод</div> </div>";

}

?>

</html>

**Удаление записей из БД**

***f\_delete.html***

<html>

<head>

<body>

<h1 align='center'>Введите имя, которое необходимо удалить</h1><BR><BR>

<form name="del\_form" action="delete.php" method="POST">

<table border='2' align='center'>

<tr>

<td>Удаление записи: <br /><input type="edit" name="del\_user"><br /><br /> </td>

</tr>

<tr>

<td><input type="submit" value="Удалить"> </td>

</tr>

</form>

</table>

</body>

</head>

</html>

***Delete.php***

<html>

<head>

<meta http-equiv="refresh" content="3;db\_connect.php">

</head>

<?

$user\_y=$\_POST['del\_user'];

//Connect to DB

$connect = mysql\_connect("localhost","user\_1", "12345");

mysql\_select\_db("site\_new",$connect);

//End connect

$result = mysql\_query("DELETE FROM userlist WHERE id=$user\_y");

if ($result==true)

{

echo "<div align=center> <div class=success>Данные успешно удалены</div> </div>";

}

else

{ echo "<div align=center> <div class=error>Ошибка при удалении данных</div> </div>";

}

mysql\_close();

?>

</html>

***Форма изменения существующей записи:***

***f\_update.php***

<h1 align='center'>Заполните свободные поля, изменив необходимую запись</h1>

<body>

<form method='post' action="update.php">

<table border='0' align='center'>

<tr>

<td>Номер:<br/><input type="edit" name="id\_up"><br/></td>

</tr>

<tr>

<td>Имя:<br/><input type="edit" name="user\_up"><br/></td>

</tr>

<tr>

<td>пароль:<br/><input type="edit" name="pass\_up"><br/></td>

</tr>

<tr>

<td><input type="submit" value="Изменить"></td>

</tr>

</form>

<br>

<br>

<br>

</body>

<?

$connect = mysql\_connect("localhost", "user\_1", "12345")

or die("База данных не подключена");

mysql\_select\_db("site\_new",$connect);

$sql = "SELECT \* FROM userlist";

$result = mysql\_query($sql) or die(mysql\_error());

$table = "<table>";

$table .= "<tr>";

$table .= "<td>Номер</td>";

$table .= "<td>Имя</td>";

$table .= "<td>Пароль</td>";

$table .= "</tr>";

while (list($id,$user,$pass) = mysql\_fetch\_array($result))

{

$table .= "<tr>";

$table .= "<td>".$id."</td>";

$table .= "<td>".$user."</td>";

$table .= "<td>".$pass."</td>";

$table .= "<td style='height:100px;'></td>";

$table .= "</tr>";

}

$table .= "</table> ";

echo $table;

mysql\_close();

?>

***Update.php***

<html>

<head>

<meta http-equiv="refresh" content="3; db\_connect.php">

</head>

<?

$id\_new=$\_POST['id\_up'];

$user\_new=$\_POST['user\_up'];

$pass\_new=$\_POST['pass\_up'];

if (trim($id\_new) and trim($user\_new) and trim($pass\_new))

{

$connect = mysql\_connect("localhost","user\_1", "12345");

mysql\_select\_db("site\_new",$connect);

//End connect

$result = mysql\_query("UPDATE userlist SET user='$user\_new', pass='$pass\_new' WHERE id='$id\_new'");

if ($result)

{

echo "<div align=center> <div class=success>Данные успешно обновлены</div> </div>";

}

else

{

echo "<div align=center> <div class=error>Ошибка при обновлении данных</div> </div>";

}

//Disconnect

mysql\_close();

}

?>

 Пример 1. Пример работы расширения MySQL

<?php

*/\* Соединение, выбор БД \*/*

*$link = mysql\_connect("mysql\_host", "mysql\_user", "mysql\_password")*

*or die("Could not connect");*

*print "Connected successfully";*

*mysql\_select\_db("my\_database") or die("Could not select database");*

*/\* Выполнение SQL query \*/*

*$query = "SELECT \* FROM my\_table";*

*$result = mysql\_query($query) or die("Query failed");*

*/\* Печать результатов в HTML \*/*

*print "<table>\n";*

*while ($line = mysql\_fetch\_array($result, MYSQL\_ASSOC)) {*

*print "\t<tr>\n";*

*foreach ($line as $col\_value) {*

*print "\t\t<td>$col\_value</td>\n";*

*}*

*print "\t</tr>\n";*

*}*

*print "</table>\n";*

*/\* Освобождение resultset \*/*

*mysql\_free\_result($result);*

*/\* Закрытие соединения \*/*

*mysql\_close($link); ?>*

***Примеры исходных файлов на PHP:***

Рассмотрим примеры скриптов для добавления, удаления, редактирования и поиска данных в базе данных со следующими полями:

* Имя - user\_name
* телефон - phone
* фотография - img (имя файла, который находится в каталоге foto/)

***Скрипт обработки загружаемых данных.***

Перед подключение к БД нужно получить переменные посланные методом GET

*$stud=$\_GET['student'];*

*$name=$\_GET['username'];*

*<?php*

*/\* Определяем значения переменным \*/*

*$hostname="localhost";*

*$username="Ваш\_логин";*

*$password="Ваш\_пароль";*

*/\* Имя базы данных \*/*

*$dbName="Имя\_вашей\_базы\_данных";*

*/\* Имя таблицы \*/*

*$usertable="Имя\_таблицы";*

*/\* Создать соединение с MySql\*/*

*MYSQL\_CONNECT($hostname,$username,$password) OR DIE("Не могу подсоединиться");*

*/\* Выбор БД \*/*

*MYSQL\_SELECT\_DB($dbName) or die("Не могу выбрать БД");*

*/\* Введение информации в БД \*/*

*$query="INSERT INTO $usertable VALUES ('$user\_name','$phone','$img')";*

*$result=MYSQL\_QUERY($query);*

*/\* Закрыть соединение \*/*

*MYSQL\_CLOSE();*

*print "Запись введена в БД! <br>";*

*?>*

***Скрипт обработки поискового запроса***

*<?php*

*$hostname="localhost";*

*$username="Ваш\_логин";*

*$password="Ваш\_пароль";*

*$dbName="Имя\_вашей\_базы\_данных";*

*$usertable="Имя\_таблицы";*

*/\* Создать соединение с MySql\*/*

*mysql\_connect($hostname,$username,$password) or die("No connect");*

*mysql\_select\_db("$dbName") or die("No select DB");*

*/\* Выбрать людей, имена которых начинаются на $usr\_name \*/*

*$query="select \* from $usertable where user\_name like '$user\_name%'";*

*$result=mysql\_query($query);*

*/\* Сколько нашлось таких сотрудников? \*/*

*$number=mysql\_numrows($result);*

*/\* Распечатка выбранных сотрудников \*/*

*$i=0;*

*if ($number == 0)*

*{*

*print "Нет данных о $user\_name в БД <br>";*

*}*

*elseif ($number > 0)*

*{*

*while ($i < $number)*

*{*

*/\* Присвоение переменным значений полей user\_name, phone, img для i-й*

*строки \*/*

*$user\_name=mysql\_result($result,$i,"user\_name");*

*$phone=mysql\_result($result,$i,"phone");*

*$img=mysql\_result($result,$i,"img");*

*/\* Вывод найденных значений \*/*

*print "<br><img src='foto/$img' height=130><br>";*

*print "<u> Имя:</u> <br>";*

*print "$user\_name <br><br>";*

*print "<u> Телефон: </u><br>";*

*print "$phone <br><br>";*

*$i++;*

*print "<hr>";*

*}*

*}*

*?>*

***Скрипт для удаления данных***

*<?php*

*$hostname="localhost";*

*$username="Ваш\_логин";*

*$password="Ваш\_пароль";*

*$dbName="Имя\_вашей\_базы\_данных";*

*$usertable="Имя\_таблицы";*

*mysql\_connect($hostname,$username,$password) or die("No connect");*

*mysql\_select\_db("$dbName") or die("No select DB");*

*/\* Удаление записи для поля user\_name \*/*

*$query="delete from $usertable where user\_name='$user\_name'";*

*$result=mysql\_query($query);*

*print "Запись удалена <br>";*

*?>*

***Скрипт редактирования данных***

*<?php*

*$hostname="localhost";*

*$username="Ваш\_логин";*

*$password="Ваш\_пароль";*

*$dbName="Имя\_вашей\_базы\_данных";*

*$usertable="Имя\_таблицы";*

*mysql\_connect($hostname,$username,$password) or die("No connect");*

*mysql\_select\_db("$dbName") or die("No select DB");*

*/\* Выбор записи по полю user\_name \*/*

*$query="select \* from $usertable where user\_name='$user\_name'";*

*$result=mysql\_query($query);*

*/\* Определение существования выбранной записи \*/*

*$number=mysql\_numrows($result);*

*if ($number==0)*

*{*

*print "Нет такой записи!<br>";*

*}*

*else*

*{*

*/\* Если запись выбрана из базы данных, вывести все ее поля в форму для*

*редактирования и удалить ее \*/*

*$phone=mysql\_result($result,0,"phone");*

*$img=mysql\_result($result,0,"img");*

*print "<img src='foto/$img' height=130><br>";*

*/\* Удаление записи \*/*

*$query="delete from $usertable where user\_name='$user\_name'";*

*$result=mysql\_query($query);*

*/\* Создание формы для редактирования \*/*

*print "<form method=post action='form5.php'>";*

*print "<br><b>Введите Ваше имя:</b>";*

*print "<br><input name='user\_name' value='$user\_name' size=30>";*

*print "<br><br><b>Введите номер вашего телефона:</b>";*

*print "<br><input name='phone' value='$phone' size=10>";*

*print "<br><br><b>Введите имя фото лат.буквами:</b>";*

*print "<br><input name='img' value='$img' size=20>";*

*print "<br><br><input type='submit' value='Зарегистрировать'>";*

*print "</form>";*

*}*

*?>*

***Скрипт для загрузки отредактированых данных***

*<?php*

*hostname="localhost";*

*$username="Ваш\_логин";*

*$password="Ваш\_пароль";*

*$dbName="Имя\_вашей\_базы\_данных";*

*$usertable="Имя\_таблицы";*

*/\* Создать соединение \*/*

*MYSQL\_CONNECT($hostname,$username,$password) OR DIE("Не могу подсоединиться");*

*/\* Выбор БД \*/*

*MYSQL\_SELECT\_DB($dbName) or die("Не могу выбрать БД");*

*/\* Введение информации в БД \*/*

*$query="INSERT INTO $usertable VALUES ('$user\_name','$phone',''$img')";*

*$result=MYSQL\_QUERY($query);*

*/\* Закрыть соединение \*/*

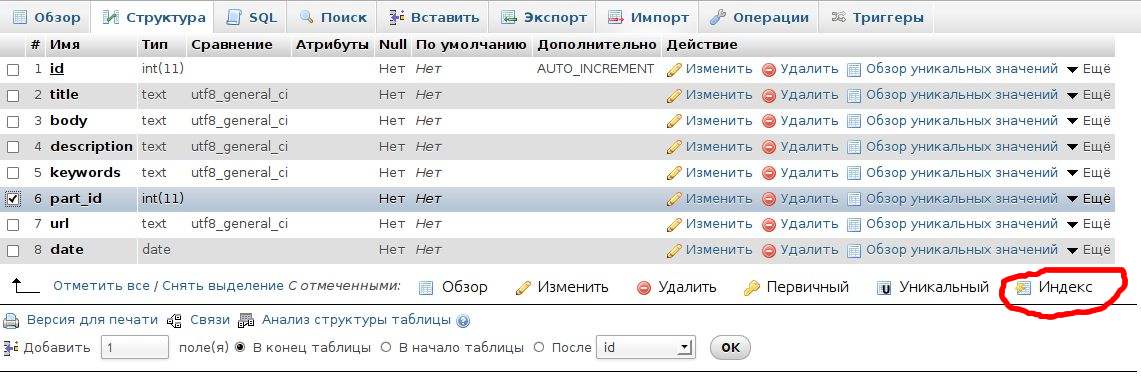
*MYSQL\_CLOSE();*

*print "Запись введена в БД! <br>";*

*?>*

**Организация связей между таблицами базы данных в PHPMyAdmin**

Для того, чтобы связать таблицы по полям, необходимо сначала **добавить в индекс**связываемые поля: В **phpmyadmin** выбираем таблицу, выбираем режим структуры, выделяем поле, для которого будем делать внешнюю связь и кликаем Индекс.

[](https://folkprog.net/assets/images/index.jpg)

Обратите внимание на разницу между "Индекс" и "Уникальный". Уникальный индекс можно

использовать, например, до поля id, то есть там, где значения не повторяются.

Это же действие можно сделать с помощью **SQL**-запроса:

**ALTER** **TABLE** `table\_name` **ADD** INDEX ( `field\_name` ) ;

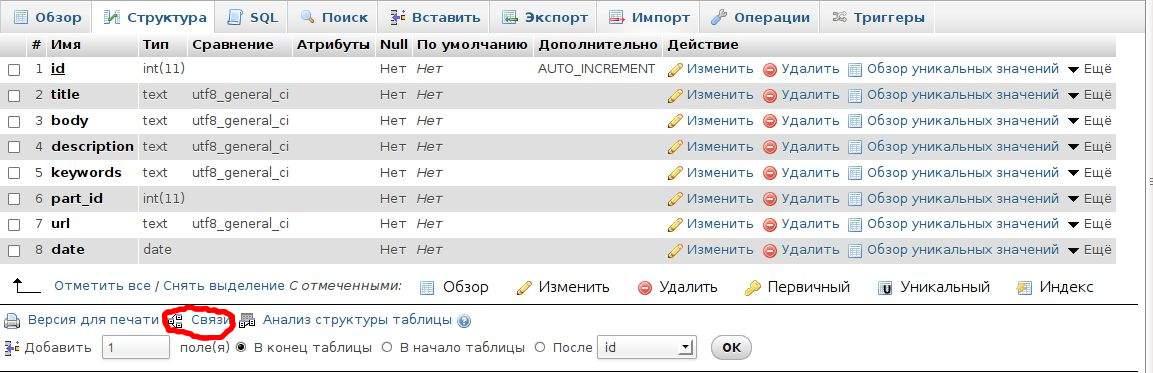
Аналогично добавляем индекс (только в моем случае теперь уже уникальный или первичный) для таблицы, на которую ссылаемся, для поля id. Поскольку поле id у меня идентификатор, для него делаем первичный ключ. Уникальный ключ мог бы понадобится для других уникальных полей.

[](https://folkprog.net/assets/images/index_uniqual.jpg)

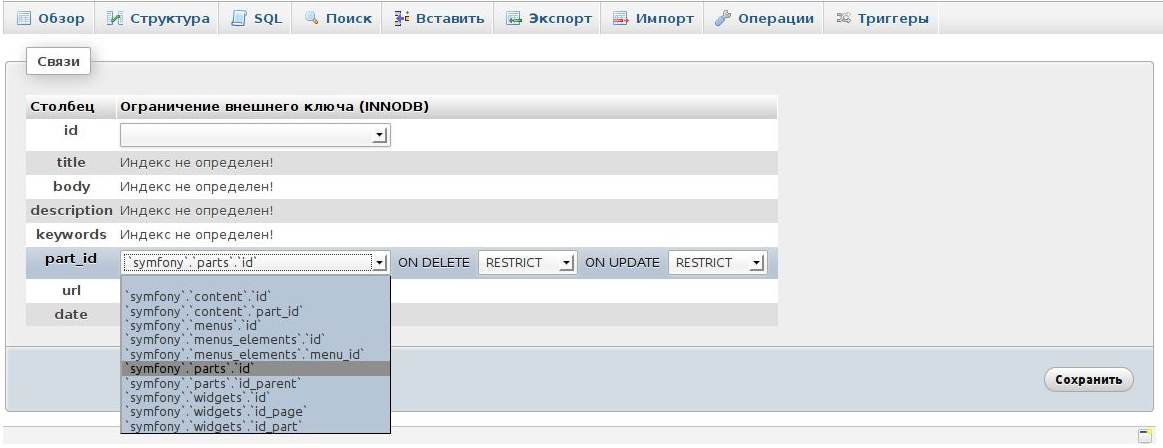
С помощью **SQL**-запроса:

**ALTER** **TABLE** `table\_name` **ADD** **UNIQUE** (`field\_name`);

Теперь осталось только **связать таблицы**. Для этого кликаем внизу на пункт Связи:

[](https://folkprog.net/assets/images/connections.jpg)

Теперь для доступных полей (а доступны только проиндексированные поля) выбираем связь с внешними таблицами и действия при изменении записей в таблицах:

[](https://folkprog.net/assets/images/connections_with_tables.jpg)

Через **SQL**-запрос:

**ALTER** **TABLE** `table\_name` **ADD** **FOREIGN** **KEY** ( `field\_in\_table\_name\_which\_need\_connect` ) **REFERENCES** `outer\_table\_to\_connect` (`outer\_field`) **ON** **DELETE** **RESTRICT** **ON** **UPDATE** **RESTRICT** ;

**Задание 2:**

1. Построить физическую схему БД.
2. Создать скрипт для подключения и создания одной из таблиц, а также добавления тестовых записей в таблицу.
3. Создать страницу и php-скрипт для выполнения одного из действий в соответствии с номером варианта (реализовать можно это на одной странице):

* добавления новой записи в таблицу;
* редактирования записи в таблице;
* удаления записи из таблицы.

**Дополнительное задание:**

* создайте базу данных «Поликлиника», состоящую из 2-3 таблиц;
* организуйте связи между таблицами БД;
* заполните таблицы информацией;
* отобразите данные таблицы на странице;
* осуществите выборку данных из БД.

**Изменение данных в БД**

*Цель:* Вносить изменения записей в базе данных.

Пусть в базе данных есть такая таблица workers:

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **name** | **age** | **salary** |
| 1 | Коля | 23 | 400 |
| 2 | Вася | 24 | 500 |
| 3 | Петя | 25 | 600 |

Давайте выведем ее в таком виде в браузер.

Для этого средствами PHP нам надо сформировать следующий HTML код:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

</tr>

<tr>

<td>1</td>

<td>Коля</td>

<td>23</td>

<td>400</td>

</tr>

<tr>

<td>2</td>

<td>Вася</td>

<td>24</td>

<td>500</td>

</tr>

<tr>

<td>3</td>

<td>Петя</td>

<td>25</td>

<td>600</td>

</tr>

</table>

Каким образом мы это сделаем: часть HTML кода мы наберем вручную, а часть за нас сформирует PHP.

Вручную мы наберем статическую часть HTML - тег table и первую tr с заголовками таблицы.

А вот собственно ряды таблицы пусть сформирует PHP, взяв данные из БД.

Наберем статическую часть HTML кода и подготовим в нем место для вставки PHP:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

</tr>

<?php

*//тут будет PHP код, который сформирует эту часть таблицы*

?>

</table>

Наш PHP код должен отправить запрос к базе данных, достать массив работников, затем сформировать из него соответствующее количество tr с td-шками.

Давайте достанем все работников из таблицы workers и запишем их в массив **$data** (пусть подключение к БД выполнено где-то выше, не будем его записывать для краткости):

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

</tr>

<?php

$query = "SELECT \* FROM workers";

$result = mysqli\_query($link, $query) or die( mysqli\_error($link) );

for ($data = []; $row = mysqli\_fetch\_assoc($result); $data[] = $row);

var\_dump($data);

?>

</table>

Полученный нами массив будет выглядеть так:

[

['id' => '1', 'name' => 'Коля', 'age' => '23', 'salary' => '400'],

['id' => '2', 'name' => 'Вася', 'age' => '24', 'salary' => '500'],

['id' => '3', 'name' => 'Петя', 'age' => '25', 'salary' => '600'],

]

Давайте переберем его циклом foreach, сформировав при этом tr-ки и td-шки:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

</tr>

<?php

$query = "SELECT \* FROM workers";

$result = mysqli\_query($link, $query) or die(mysqli\_error($link));

for ($data = []; $row = mysqli\_fetch\_assoc($result); $data[] = $row);

$result = '';

foreach ($data as $elem) {

$result .= '<tr>';

$result .= '<td>' . $elem['id'] . '</td>';

$result .= '<td>' . $elem['name'] . '</td>';

$result .= '<td>' . $elem['age'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '</tr>';

}

echo $result;

?>

</table>

Если запустить этот код (не забыв про подключение к БД), то на экран выведется соответствующая таблица.

При этом, конечно же, наша HTML таблица будет формироваться из тех данных, которые есть в текущий момент в БД. То есть если там будет больше записей - то и в браузере будет выведено больше записей.

Давайте теперь сделаем добавление нового работника с помощью формы.

Давайте сделаем HTML код формы добавления:

<form action="" method="POST">

<input name="name">

<input name="age">

<input name="salary">

<input type="submit" value="добавить работника">

</form>

А теперь напишем PHP код, который будет формировать INSERT запрос для сохранения данных из формы в базу данных:

<?php

$name = **$\_POST**['name'];

$age = **$\_POST**['age'];

$salary = **$\_POST**['salary'];

$query = "INSERT INTO workers SET name='**$name**', age='**$age**', salary='**$salary**'";

mysqli\_query($link, $query) or die(mysqli\_error($link));

?>

Давайте теперь совместим нашу форму и код, обрабатывающий данные с нее в одном файле.

Напомню, что в этом случае наш скрипт выполнится два раза: первый раз пользователь зайдет на страницу, заполнит форму и отправит ее. После отправления мы попадем на эту же страницу и скрипт начнет выполнятся сначала, но уже будут доступны данные из формы.

Эти данные будут хранится в $\_POST, если форма, как у нас, отправлена методом POST.

Наш PHP код для сохранения должен выполнится после того, как форма была отправлена и не должен выполнятся при первом заходе пользователя на страницу.

Добиться этого мы можем с помощью ифа. Например, можно спросить, не пустой ли POST и, если не пустой - только тогда начинать выполнять наш код для сохранения, вот так:

<?php

if (!empty(**$\_POST**)) {

$name = **$\_POST**['name'];

$age = **$\_POST**['age'];

$salary = **$\_POST**['salary'];

$query = "INSERT INTO workers SET name='**$name**', age='**$age**', salary='**$salary**'";

mysqli\_query($link, $query) or die(mysqli\_error($link));

}

?>

Давайте совместим наш PHP и форму:

<?php

if (!empty(**$\_POST**)) {

$name = **$\_POST**['name'];

$age = **$\_POST**['age'];

$salary = **$\_POST**['salary'];

$query = "INSERT INTO workers SET name='**$name**', age='**$age**', salary='**$salary**'";

mysqli\_query($link, $query) or die(mysqli\_error($link));

}

?>

<form action="" method="POST">

<input name="name">

<input name="age">

<input name="salary">

<input type="submit" value="добавить работника">

</form>

Форму под PHP кодом, но ее можно разместить где угодно - ведь наш файл со скриптом выполняется два раза и поэтому разницы нет, где что будет размещено.

В общем код, решающий поставленную задачу, написан. Его можно разместить на отдельной странице, а можно совместить с нашей HTML таблицей работников.

Первый вариант у нас уже реализован - можете потестировать его, разместив этот код в отдельном файле и проверив его работу (не забудьте про подключение к базе данных, которое я опускаю для краткости).

Давайте теперь совместим вывод таблицы работников с добавлением нового.

Вспомним код, который делает вывод таблицы и удаление:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

<th>delete</th>

</tr>

<?php

*// Удаление по id (до получения!):*

if (isset(**$\_GET**['del'])) {

$del = **$\_GET**['del'];

$query = "DELETE FROM workers WHERE id=**$del**";

mysqli\_query($link, $query) or die(mysqli\_error($link));

}

*// Получение всех работников:*

$query = "SELECT \* FROM workers";

$result = mysqli\_query($link, $query) or die(mysqli\_error($link));

for ($data = []; $row = mysqli\_fetch\_assoc($result); $data[] = $row);

*// Вывод на экран:*

$result = '';

foreach ($data as $elem) {

$result .= '<tr>';

$result .= '<td>' . $elem['id'] . '</td>';

$result .= '<td>' . $elem['name'] . '</td>';

$result .= '<td>' . $elem['age'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td><a href="?del=' . $elem['id'] . '">удалить</a></td>';

$result .= '</tr>';

}

echo $result;

?>

</table>

Давайте совместим его с нашим кодом, который делает добавление нового работника. При этом форму добавления поставим под таблицу (ее можно поставить куда угодно), а сам PHP код добавления разместим до получения всех работников, чтобы сначала добавлялся новый и потом получались все работники вместе с новым добавленным.

Итак, вот наш код:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

<th>delete</th>

</tr>

<?php

*// Сохранение нового (до получения!):*

if (!empty(**$\_POST**)) {

$name = **$\_POST**['name'];

$age = **$\_POST**['age'];

$salary = **$\_POST**['salary'];

$query = "INSERT INTO workers SET name='**$name**', age='**$age**', salary='**$salary**'";

mysqli\_query($link, $query) or die(mysqli\_error($link));

}

*// Удаление по id (до получения!):*

if (isset(**$\_GET**['del'])) {

$del = **$\_GET**['del'];

$query = "DELETE FROM workers WHERE id=**$del**";

mysqli\_query($link, $query) or die(mysqli\_error($link));

}

*// Получение всех работников:*

$query = "SELECT \* FROM workers";

$result = mysqli\_query($link, $query) or die(mysqli\_error($link));

for ($data = []; $row = mysqli\_fetch\_assoc($result); $data[] = $row);

*// Вывод на экран:*

$result = '';

foreach ($data as $elem) {

$result .= '<tr>';

$result .= '<td>' . $elem['id'] . '</td>';

$result .= '<td>' . $elem['name'] . '</td>';

$result .= '<td>' . $elem['age'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td><a href="?del=' . $elem['id'] . '">удалить</a></td>';

$result .= '</tr>';

}

echo $result;

?>

</table>

<form action="" method="POST">

<input name="name">

<input name="age">

<input name="salary">

<input type="submit" value="добавить работника">

</form>

**Удаление данных из БД**

*Цель:*научится удаоять зписи из базы данных

Давайте модифицируем задачу предыдущего урока, добавив возможность удаления работников.

Для этого добавим в таблицу еще одну колонку, в которой для каждого работника будет размещаться ссылка на удаление работника:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

<th>delete</th>

</tr>

<tr>

<td>1</td>

<td>Коля</td>

<td>23</td>

<td>400</td>

<td><a href="">удалить</a></td>

</tr>

<tr>

<td>2</td>

<td>Вася</td>

<td>24</td>

<td>500</td>

<td><a href="">удалить</a></td>

</tr>

<tr>

<td>3</td>

<td>Петя</td>

<td>25</td>

<td>600</td>

<td><a href="">удалить</a></td>

</tr>

</table>

Сделаем так, чтобы при переходе по ссылке мы попадали на ту же страницу браузера, но отправляя при этом GET запрос с id работника, которого мы хотим удалить:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

<th>delete</th>

</tr>

<tr>

<td>1</td>

<td>Коля</td>

<td>23</td>

<td>400</td>

<td><a href="?del=1">удалить</a></td>

</tr>

<tr>

<td>2</td>

<td>Вася</td>

<td>24</td>

<td>500</td>

<td><a href="?del=2">удалить</a></td>

</tr>

<tr>

<td>3</td>

<td>Петя</td>

<td>25</td>

<td>600</td>

<td><a href="?del=3">удалить</a></td>

</tr>

</table>

Как это будет работать: если мы перейдем, например, по ссылке для работника "Коля", то отправим GET запросом параметр del со значением 1, соответствующим id Коли в таблице базы данных.

В коде PHP мы можем получить id работника для удаления, обратившись к $\_GET['del'] и затем удалить его, вот так:

<?php

$del = **$\_GET**['del']; *// получим id для удаления*

$query = "DELETE FROM workers WHERE id=**$del**"; *// сформируем запрос на удаление*

mysqli\_query($link, $query) or die(mysqli\_error($link)); *// удалим*

?>

Так как мы не всегда выполняем операцию удаления, то GET параметра может и не быть в адресной строке. Поэтому давайте проверим его наличие с помощью функции isset - и только, если параметр есть - будем выполнять удаление:

<?php

if (isset(**$\_GET**['del'])) {

$del = **$\_GET**['del'];

$query = "DELETE FROM workers WHERE id=**$del**";

mysqli\_query($link, $query) or die(mysqli\_error($link));

}

?>

Давайте теперь вспомним код для вывода данных в виде HTML таблицы, полученный нами в предыдущем уроке:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

</tr>

<?php

$query = "SELECT \* FROM workers";

$result = mysqli\_query($link, $query) or die( mysqli\_error($link) );

for ($data = []; $row = mysqli\_fetch\_assoc($result); $data[] = $row);

$result = '';

foreach ($data as $elem) {

$result .= '<tr>';

$result .= '<td>' . $elem['id'] . '</td>';

$result .= '<td>' . $elem['name'] . '</td>';

$result .= '<td>' . $elem['age'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '</tr>';

}

echo $result;

?>

</table>

Добавим в таблицу еще одну ячейку со ссылкой на удаление (пока без GET запроса):

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

<th>delete</th>

</tr>

<?php

$query = "SELECT \* FROM workers";

$result = mysqli\_query($link, $query) or die( mysqli\_error($link) );

for ($data = []; $row = mysqli\_fetch\_assoc($result); $data[] = $row);

$result = '';

foreach ($data as $elem) {

$result .= '<tr>';

$result .= '<td>' . $elem['id'] . '</td>';

$result .= '<td>' . $elem['name'] . '</td>';

$result .= '<td>' . $elem['age'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td><a href="">удалить</a></td>';

$result .= '</tr>';

}

echo $result;

?>

</table>

Давайте теперь сделаем так, чтобы при переходе по ссылке передавался GET запрос на удаление.

Так как наши tr-ки формируются в цикле, мы не можем просто вручную проставить номера id для удаления в GET запрос. Пусть это сделает PHP автоматически.

Сам id работника хранится в $elem['id'] - подставим это значение в GET запрос в href ссылки, вот так:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

<th>delete</th>

</tr>

<?php

$query = "SELECT \* FROM workers";

$result = mysqli\_query($link, $query) or die( mysqli\_error($link) );

for ($data = []; $row = mysqli\_fetch\_assoc($result); $data[] = $row);

$result = '';

foreach ($data as $elem) {

$result .= '<tr>';

$result .= '<td>' . $elem['id'] . '</td>';

$result .= '<td>' . $elem['name'] . '</td>';

$result .= '<td>' . $elem['age'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td><a href="?del=' . $elem['id'] . '">удалить</a></td>';

$result .= '</tr>';

}

echo $result;

?>

</table>

Теперь при переходе по ссылке будет происходить передача GET параметра с id работника, которого мы хотим удалить. Но самого удаления пока не будет, так как мы не выполняем SQL запрос на удаление.

Давайте добавим код для удаления, полученный нами в начале урока. Учтите, что его нужно добавлять **до получения** работников из БД, чтобы при удалении работник сначала удалился из базы, а уже потом мы получили оставшихся и вывели их на экран:

<table>

<tr>

<th>id</th>

<th>name</th>

<th>age</th>

<th>salary</th>

<th>delete</th>

</tr>

<?php

*// Удаление по id (до получения!):*

if (isset(**$\_GET**['del'])) {

$del = **$\_GET**['del'];

$query = "DELETE FROM workers WHERE id=**$del**";

mysqli\_query($link, $query) or die(mysqli\_error($link));

}

*// Получение всех работников:*

$query = "SELECT \* FROM workers";

$result = mysqli\_query($link, $query) or die(mysqli\_error($link));

for ($data = []; $row = mysqli\_fetch\_assoc($result); $data[] = $row);

*// Вывод на экран:*

$result = '';

foreach ($data as $elem) {

$result .= '<tr>';

$result .= '<td>' . $elem['id'] . '</td>';

$result .= '<td>' . $elem['name'] . '</td>';

$result .= '<td>' . $elem['age'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td>' . $elem['salary'] . '</td>';

$result .= '<td><a href="?del=' . $elem['id'] . '">удалить</a></td>';

$result .= '</tr>';

}

echo $result;

?>

</table>

Если запустить этот код, то в браузере мы увидим список работников. Если затем перейти по ссылке для удаления какого-либо работника, страница браузера перезагрузится (т.к. мы перешли по ссылке), работник удалится из базы и в таблице его уже не будет.

Еще раз: удаление следует размещать до получения работников из БД, иначе вы сначала получите всех работников вместе с тем, которого хотели удалить, только затем удалите его в базе, но в HTML таблице работник не удалится. Его удаление произойдет только после перезагрузки страницы. Учтите это и не совершайте такой ошибки.