|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | Работа 1. Простейшие PHP-программы |  |  |  |  |  | | --- | |  | |  | | Упражнение 1. Начальный и конечный теги PHP  Синтаксический анализатор PHP распознает несколько различных тегов начала и конца PHP-кода. Он пытается выполнить все, что расположено между этими тэгами. В присоединной ниже таблице показаны три основных типа тэгов начала и конца PHP-кода.  Основные тэги начала и конца PHP-кода   |  |  | | --- | --- | | Открывающий (начальный) тэг | Закрывающий (конечный) тэг | | <?php | ?> | | <? | ?> | | <script language="php"> | </script> |  * Откройте любой текстовый редактор, например Блокнот. * Введите следующий код, в котором используется первый тип тегов:   http://activ.polytech.kz/images/stories/labs/php/01/php01.1.gif   * Добавьте код, в котором используется второй тип тэгов:   http://activ.polytech.kz/images/stories/labs/php/01/php01.2.gif  Добавьте код, в котором используется третий тип тэгов:  http://activ.polytech.kz/images/stories/labs/php/01/php01.3.gif   * Сохраните созданный файл. При сохранении в Блокноте заключите название файла в кавычки, чтобы автоматически не было добавлено расширение .txt. Скопируйте его в выделенную папку на Web-сервере. * Запустите браузер, в поле **Адрес** введите:  **http://localhost/php/test01.php** * В рабочем окне появится результат работы скрипта:   http://activ.polytech.kz/images/stories/labs/php/01/php01.4.gif  Упражнение 2. Совместное использование кодов  В предыдущем упражнении созданный файл test01.php состоял из трех фрагментов PHP-кода, каждый из которых печатал определенный текст HTML. В этом упражнении мы создадим сценарий, в котором PHP-код будет находиться внутри HTML, и вы увидите, что два типа кодов могут существовать в одном документе.  Откройте Блокнот.  Введите следующий код HTML:  http://activ.polytech.kz/images/stories/labs/php/01/php01.5.gif   * Далее введите следующий код PHP:   http://activ.polytech.kz/images/stories/labs/php/01/php01.6.gif   * Следом добавьте завершающий код HTML:   http://activ.polytech.kz/images/stories/labs/php/01/php01.7.gif   * Сохраните файл с именем **firstscript.php**и скопируйте его в выделенную папку на Web-сервере. * Запустите браузер, в поле **Адрес** укажите URL скрипта. * Выберите в браузере команду **Вид | В виде HTML.**На экране появится рабочее окно текстового редактора Блокнот с исходным кодом HTML-Документа.   Как видно, исходный код HTML-документа не содержит кода PHP, а только результат его работы.  Инструкция *echo*  Описанный в предыдущем примере код PHP содержит одну инструкцию, состоящую из трех элементов: команды (echo), строки (<p><i>Привет всем! Я использую PHP!</i></p>) и признака конца инструкции (;). Познакомимся с командой **echo** поближе, потому что она наиболее часто используется для вывода информации. Например, в предыдущем примере, -  для выдачи результирующего кода HTML:  <p><i>Привет всем! Я использую PHP!</i></p>  Упражнение 3. Делаем ошибки  Признак конца инструкции или символ (;) является обязательным в конце любой команды. Признак конца инструкции сообщает синтаксическому анализатору PHP, что параметры закончились, и следует перейти к следующей инструкции.  Если вы не закончите инструкцию точкой с запятой, то синтаксический анализатор PHP выдаст сообщение об ошибке.  Откройте Блокнот  Введите следующий текст:  http://activ.polytech.kz/images/stories/labs/php/01/php01.8.gif   * Сохраните этот файл с именем errorscript.php и скопируйте его в выделенную папку на Web-сервере. * Запустите браузер, в поле **Адрес** введите URL скрипта и нажмите Enter. В рабочем окне браузера появится сообщение об ошибке:   http://activ.polytech.kz/images/stories/labs/php/01/php01.9.gif  Как видно из рисунка, синтаксический анализатор выдал сообщение об ошибке в 8-й строке файла. Если внимательно посмотреть, то можно заметить, что в строке 7 отсутствует признак конца инструкции, а строка 8 начинается с новой команды.  Эту ошибку легко исправить.   * Откройте с помощью блокнота файл **errorscript.php**. * В конце 7-й строки добавьте признак завершения инструкции(;). * Сохраните файл и скопируйте его в выделенную папку на Web-сервере. * Если в браузере еще открыт этот файл, то нажмите клавишу F5 или кнопку **Обновить** на панели инструментов. * В рабочем окне браузера должен появиться результат работы скрипта:   http://activ.polytech.kz/images/stories/labs/php/01/php01.10.gif  Упражнение 4. Маскирование специальных символов в сценарии  Теперь мы познакомимся с тем, как маскировать знак "кавычки", который вы хотите вывести на экран. Когда вы используете кавычки внутри других кавычек, то внутренняя пара должна быть замаскирована, скрыта от внешней пары с помощью символа маскирования - обратной косой линии (\).  В следующем примере показано, что произойдет, если не будет использован символ обратной косой линии, и как исправить ошибку.   * Откройте Блокнот. * Введите следующий текст HTML:   http://activ.polytech.kz/images/stories/labs/php/01/php01.11.gif   * Сохраните файл под именем **errorscript2.php**и скопируйте его в выделенную папку на Web-сервер. * Запустите браузер и набелите URL файла. В рабочем окне появится сообщение об ошибке:   http://activ.polytech.kz/images/stories/labs/php/01/php01.12.gif  Здесь внутренняя пара кавычек ("правильно") используется внутри внешней пары кавычек без символа маскирования, что вызывает сообщение об ошибке. Эта ошибка легко исправляется. Чтобы исправить ошибку выполните следующие действия:   * Откройте с помощью Блокнота файл **errorscript2.php** * Измените в строке 8 текст на:   **echo "<p>Я думаю это действительно \"правильно\"!</p>";**   * Сохраните файл и скопируйте его в выделенную папку на Web-сервере. * Просмотрите результат выполнения с помощью браузера.   http://activ.polytech.kz/images/stories/labs/php/01/php01.13.gif  Теперь перед внутренними кавычками стоит символ маскирования (\). Синтаксический анализатор PHP пропустит их, зная, что символы, следующие за символом маскирования, должны быть просто напечатаны.  Упражнение 5. Комментирование программ  Ввод комментариев в HTML-документ поможет вам отслеживать отдельные фрагменты большого документа. С помощью комментариев вы можете вставлять в текст программы собственные примечания, которые помогут вам в процессе ее разработки и отладки.  HTML-комментарии игнорируются браузером и содержатся между тэгами <!-- и -->. Например, следующий комментарий напомнит вам, что ниже расположены коды вашего логотипа:  **<!-- Логотип расположен здесь -->**  В PHP тоже используются комментарии, которые игнорируются синтаксическим анализатором. PHP-комментариям обычно предшествует двойной слеш, например:  **// Это комментарии в сценарии PHP**  Но могут быть использованы и другие типы комментариев, такие как:  **# Это стиль комментария командного языка операционной системы**  и  **/\* Это С-стиль комментария, который**  **будет расположен на двух строках \*/**  Создадим сценарий, содержащий различные типы комментариев.   * Откройте Блокнот * Введите следующий текст   http://activ.polytech.kz/images/stories/labs/php/01/php01.14.gif  Сохраните файл с именем **comments.php**и скопируйте его в выделенную папку на Web-сервере.  Просмотрите результат выполнения скрипта браузером. В рабочем окне браузера вы не увидите абсолютно ничего:  http://activ.polytech.kz/images/stories/labs/php/01/php01.15.gif  Пустая страница получилась потому, что все, что вы набрали, было HTML-комментарием, который игнорируется браузером. Поскольку PHP-комментарии также игнорируются синтаксическим анализатором PHP, то фрагмент PHP-кода не содержит никаких действительных команд.   * Выберите команду в браузере меню **Вид | В виде HTML**. На экране появится рабочее окно текстового редактора Блокнот с исходным текстом HTML-документа:   http://activ.polytech.kz/images/stories/labs/php/01/php01.16.gif  Обратите внимание на то, что исходный код HTML-документа содержит только HTML-комментарии, комментарии PHP отсутствуют. |   Работа 2. Переменные и операторы |  |  |  |

|  |
| --- |
|  |
|  |
| Знакомство с переменными и операторами  В этой лабораторной работе вы научитесь:   * определять и использовать переменные * определять и использовать операторы     1. Что такое переменная?  В переменной могут храниться значения определенного типа. После присвоения значения переменной вы можете сослаться на эту переменную в любом месте вашей программы, и это значение будет всегда оставаться неизменным, пока вы сами его не измените.  Чтобы создать переменную необходимо выполнить следующие действия:   * Придумать какое-нибудь подходящее имя. Например, если вы хотите создать переменную, которая будет хранить имя пользователя, то подходящим именем для переменной будет **username**. * Поставить символ доллара (**$**) перед именем, как показано ниже: **$username**. * Поставить знак равенства (=) после имени переменной, чтобы присвоить ей значение. И заключить значение в кавычки, как показано ниже: **$username="Иванов"** * Чтобы присвоение значения переменной стало инструкцией, завершите ее точкой с запятой (;): **$username="Иванов";**   Итак, вы узнали, что имена переменных начинаются со знака доллара ($). Имена переменных не могут начинаться с цифры, но могут их содержать, а также содержать символ подчеркивания (\_). Кроме того, имена переменных чувствительны к регистру, например, переменные **$YOURVAR** и **$yourvar** - две разные переменные.  2. Типы переменные PHP и их значений  При программировании в PHP вы будете иметь дело с двумя основными типами переменных: скалярным типом и массивами. Скалярные переменные содержат только одно значение, а массивы состоят из списка знаяений или даже нескольких списков.  В описанном примере мы создали переменную $username, которая является скалярной.  Когда вы присваиваете значение переменной, вы обычно указываете один из следующих типов значений:   * **целое** (integer) - целое число (число без дробной части). Например, 1345 или 992786. * **число с плавающей точкой** (floating-point или floats или doubles) - числа с дробной частью. Например, 1.5, 87.3446, 0.88889992; * **строка** (string) - выражение, состоящее из текста, или цифр или текста с цифрами, заключенные в парные двойные (" ") или одинарные (' ') кавычки.   Создадим простой сценарий, в котором объявим различные типы переменных, присвоим им значения и затем просто выведем их значения на экран.   * Откройте текстовый редактор и введите текст HTML:   <html> <head> <title>Печать переменных</title> </head> <body>   * Добавьте PHP-блок и создайте переменную целого типа:   <? $intVar = "9554215464";   * Создайте переменную с плавающей точкой:   $floatVar = "1542.2232235";   * Создадим строковую переменную:   $stringVar = "Это строка.";   * Добавьте команду **echo** для каждой переменной, чтобы вывести их значения на печать:   echo "<p>integer (целая): $intVar</p>"; echo "<p>float (дробная): $floatVar</p>"; echo "<p>string (строковая): $stringVar</p>";   * Закройте PHP-блок и добавьте несколько строк HTML, чтобы документ стал законченным и правильным:   ?> </body> </html>   * Сохраните документ под именем printvarscript.php и скопируйте его в выделенную папку на Web-сервере. * Запустите браузер, и вызовите этот документ. * Просмотрите результат работы программы.   3. Операторы PHP  Выше мы уже использовали один из операторов (=) для присвоения значения вашим переменным. Существуют и другие типы операторов присвоения, а также другие типы операторов вообще.  Перечислим основные типы операторов PHP и их основные функции:   * операторы присвоения - присваивают значения переменным. Могут также увеличивать или уменьшать текущее значение переменной; * арифметические операторы - это операторы сложения, вычитания, деления и умножения, которые используются, когда необходимо вычислить значения выражения; * операторы сравнения - применяются для сравнения двух переменных, результатом сравнения будет истина (true) или ложь (false); * логические операторы - проверяют истинность условия.   4. Операторы присвоения  Знак равенства (=) является основным оператором присвоения.  В приведенной ниже таблице приведены различные типы операторов присвоения и примеры их записи в сценарии.  Операторы присвоения   |  |  |  | | --- | --- | --- | | Оператор | Пример записи | Выполняемое действие | | += | $a += 3; | увеличивает текущее значение переменной на значение, находящееся справа от знака оператора. | | -= | $a -= 3; | Уменьшает текущее значение переменной на значение, находящееся справа от знака оператора | | .= | $a .= "строка"; | Объединяет (добавляет к) текущее значение со значением, находящимся справа от знака равенства. |   Создадим простой сценарий, показывающий как работают эти операторы. Этот сценарий присвоит переменным значения, затем изменит эти значения и выведет результат на экран.   * Откройте новый текстовый документ и введите текст HTML:   <html> <head> <title>Использование оператора присвоения</title> </head> <body>   * Начните PHP-блок, создайте переменную целого типа со значением 100 и выведите ее значение:   <? $origVar = 100; echo "<p>Переменной присвоено значение $origVar</p>";   * Увеличьте значение переменной на 25 и выведите результат:   $origVar += 25; echo "<p>Теперь значение переменной увеличено и составляет $origVar</p>";   * Уменьшите текущее значение переменной на 12:   $origVar -= 12; echo "<p>Теперь значение переменной уеньшено и составляет $origVar</p>";   * Соедините текущее значение с текстом и выведите результат:   $origVar .= " цыплят"; echo "<p>Окончательное значение переменной такое: $origVar</p>";   * Закройте PHP-блок и добавьте несколько строк HTML, чтобы документ стал законченным и правильным:   ?> </body> </html>   * Сохраните документ с именем assignscript.php и скопируйте его в выделенную папку на Web-сервере. Просмотрите документ в браузере.   5. Арифметические операции  **Арифметические операторы**   |  |  |  | | --- | --- | --- | | Оператор | Пример записи | Выполняемое действие | | + | $b = $a + 3; | Складывает значения | | - | $b = $a - 3; | Вычитает значения | | \* | $b = $a \* 3; | Умножает значения | | / | $b = $a / 3; | Делит значения | | % | $b = $a % 3; | Определяет остаток от целочисленного деления одной переменной на другую |   Создадим простой сценарий, иллюстрирующий работу всех этих операторов. Этот сценарий присвоит двум переменным определенные значения, произведет над ними арифметические действия и выведет результат на экран.   * С помощью текстового редактора введите текст HTML:   <html> <head> <title>Использование арифметических операторов</title> </head> <body>   * Введите блок PHP:   <?  $a = 85; $b = 24; echo "<p>Присвоены значения переменным \$a = $a и \$b = $b</p>";  $c = $a + $b; echo "<p>Сумма \$a + \$b присвоена переменной \$c и равна $c</p>";  $c = $a - $b; echo "<p>Вычитаем \$b из \$a и присваиваем переменной \$c, которая станет равной $c</p>";  $c = $a \* $c; echo "<p>Умножаем \$a и \$b и получаем $c</p>";  $c = $a / $b; echo "<p>Делим \$a на \$b и получаем $c</p>";  $c = $a % $b; echo "<p>Определяем остаток от целочисленного деления \$a на \$b и получаем $c</p>";  ?>   * Добавьте несколько строк HTML, чтобы завершить документ.   </body> </html>   * Сохраните набранный файл с именем arithmeticscript.php и скопируйте его в выделенную папку на Web-сервере.  |  | | --- | | **Примечание** | | Если вы используете символ маскирования перед знаком доллара (\$), то он будет отображен как символ, иначе начинается интерпретация его как переменной. |  * протестируйте скрипт в браузере.   6. Операторы сравнения  **Операторы сравнения**   |  |  | | --- | --- | | Оператор | Определение | | == | Равно | | != | Не равно | | > | Больше чем | | < | Меньше чем | | >= | Больше или равно | | <= | Меньше или равно |   Результатом каждой из этих операций является либо истина либо ложь.  Создадим простой сценарий для демонстрации результата выполнения некоторых операторов сравнения с помощью управления if...else и выведем результат на экран.   * С помощью текстового редактора введите код HTML:   <html> <head> <title>Использование операторов сравнения</title> </head> <body>   * Начнем PHP-блок:   <?  $a = 21; $b = 15; echo "<p>Вводим переменные \$a = $a и \$b = $b</p>";   * Внутри оператора if ... else проверим, равны ли переменные $a и $b.   if ($a == $b) {  echo "<p>Проверка 1: \$a равно с \$b</p>"; } else {  echo "<p>Проверка 1: \$a не равно с \$b</p>"; }  Обратите внимание, что выражение сравнения должно помещаться в круглые скобки. Фигурные скобки ({) и (}) служат для разделения блоков операторов внутри структуры управления.   * Теперь выполним проверку тех же операторов на неравенство и, в зависимости от результата, выведем сообщение на экран с помощью оператора echo:   if ($a != $b) {  echo "<p>Проверка 2: \$a не равно с \$b</p>"; } else {  echo "<p>Проверка 2: \$a равно с \$b</p>"; }   * Проверим, больше ли $a, чем $b и, в зависимости от результата, выведем сообщение на экран:   if ($a > $b) {  echo "<p>Проверка 3: \$a больше чем \$b</p>"; } else {  echo "<p>Проверка 3: \$a не больше чем \$b</p>"; }   * Проверим, меньше ли $a, чем $b и, в зависимости от результата, выведем сообщение на экран:   if ($a < $b) {  echo "<p>Проверка 3: \$a меньше чем \$b</p>"; } else {  else "<p>Проверка 3: \$a не меньше чем \$b</p>"; }   * Выполните аналогичные проверки с операторами (>=) и (<=). * Закройте PHP-блок и завершите документ HTML:   ?> </body> </html>   * Сохраните набранный файл с именем comparisonscript.php и скопируйте его в выделенную папку на web-сервере. * Просмотрите результат в браузере.   7. Логические операторы  Сейчас  мы рассмотрим логические операторы && (and - "и") и || (or - "или"), которые применяются для вычисления логического выражения, в котором присутствуют несколько сравнений.   * С помощью текстового редактора введите текст HTML:   <html> <head> <title>Использование логически операторов</title> </head> <body>   * Начните PHP-блок, создайте две переменных и присвойте им значения:   <?  $degrees = "95"; $hot = "да";   * Внутри оператора if ... else проверьте истинность условия "значение переменной $degrees больше 100 или значение переменной $hot равно "да". И, в зависимости от результата, выведите сообщение с помощью оператора echo.   if (($degrees > 100) || ($hot =="да")) {  echo "<p>Проверка 1: Это <b>действительно</b> жарко!</p>"; } else {  echo "<p>Проверка 1: Это терпимо</p>"; }   * Внутри оператора if ... else введите вторую проверку истинности условия "значение переменной $degrees больше 80 и значение переменной $hot равно "да". И, в зависимости от результата, выведите сообщение с помощью команды echo.   if (($degrees > 80) && ($hot =="да")) {  echo "<p>Проверка 2: Это <b>действительно</b> жарко!</p>"; } else {  echo "<p>Проверка 2: Это терпимо</p>"; }   * Закройте PHP-блок и завершите документ HTML:   ?> </body> </html>   * Сохраните набранный файл с именем logicalscript.php и скопируйте его в выделенную папку на web-сервере. * Просмотрите результат в браузере. |

|  |  |  |  |
| --- | --- | --- | --- |
| Создание форм для ввода и передачи данных на Web-сервер |  |  |  |

|  |
| --- |
|  |
|  |
| Использование переменных  В этой лабораторной работе вы научитесь:   * Использовать HTML-формы, чтобы вводить значения переменных в ваши программы; * Использовать переменные окружения     1. Получение переменных из форм  Формы HTML должны содержать следующие элементы:   * метод; * действие; * кнопку для пересылки формы (Submit).   Для примера создадим форму для сценария калькулятора. Эта форма будет содержать два поля ввода и переключатели для выбора типа расчета.   * Откройте текстовый редактор и введите текст HTML:   <html> <head> <title>Форма калькулятора</title> </head> <body>   * Чтобы начать создание формы методом post и действием, выполняющемся в сценарии calculate.php, введите заголовок формы:   <fom method = "post" action = "calculate.php">   * Чтобы создать поле ввода с надписью для первой переменной, введите:   <p>Значение 1: <input type="text" name="val1" size=10></p>   * Чтобы создать поле ввода с надписью для второй переменной, введите:   <p>Значение 2: <input type="text" name="val2" size=10></p>   * Чтобы создать кнопку с надписью "Вычислить", введите текст:   <p><input type="submit" name="submit" value="Вычислить"></p>   * Чтобы закрыть форму, введите тэг:   </form>   * Введите HTML-тэги, чтобы завершить документ:   </body> </html>   * Сохраните созданный документ под именем calculate\_form.html, скопируйте его в выделенную папку на Web-сервере и выполните его.   В окне браузера находится форма, содержащая поля ввода **Значение 1**и **Значение 2**и кнопку **Вычислить**.  При пересылке формы в ваш сценарий будут переданы две переменные - **$val1**и **$val2,** так как именно эти имена являются значениями параметра **name**, использованного при создании каждого текстового поля. Значениями этих переменных будут значения, введенные пользователем в соответствующие поля формы.  Теперь добавим к уже созданной форме переключатели, состояние которых будет определять тип операции над двумя значениями.   * Откройте с помощью текстового редактора, файл calculate\_form.html. * Перед строкой, задающей параметры кнопки **Вычислить**, введите следующий фрагмент HTML-кода:   <p>Действие:<br> <input type="radio" name="calc" value="сложить"> сложить<br> <input type="radio" name="calc" value="вычесть"> вычесть<br> <input type="radio" name="calc" value="умножить"> умножить<br> <input type="radio" name="calc" value="рзделить"> рзделить<br>   * Сохраните изменения на Web-сервере. * Обновите страницу в браузере клавишей F5.   Теперь созданная форма содержит два поля ввода Значение 1 и Значение 2, переключатель Действие и кнопку Вычислить. Кроме того, к уже существующим двум переменным ($val1 и $val2) добавилась еще одна с именем $calc, которая также будет посылаться в ваш сценарий.  Теперь создадим сценарий калькулятора.  В соответствии с выбранным в форме действием, необходимо назвать PHP-сценарий calculate. php. Цель сценария - принять два значения переменных $val1 и $val2 и произвести с ними вычисления в зависимости от значения переменной $calc.   * Создайте новый файл в текстовом редакторе. * Начните PHP-блок и подготовьте оператор if к проверке наличия трех переменных, введя следующий текст:   <? if (($val1 == "") || ($val2 == "") || ($calc == "")) {  header ("Location: http://localhost/calculate\_form.html");  exit; }  Этот оператор условия проверяет, имеют ли переданные в сценарий переменные значения, если хотя бы одна из них имеет пустое значение, то оператор header перенаправить пользователя снова на форму ввода данных. *Внимание! Удостоверьтесь, что перед началом PHP-блока нет пустых строк, пробелов или другого текста. Вы не можете использовать функцию header( ), если что-то было отправлено браузеру.*   * Начните оператор if...else для выполнения расчета, основанного на значении переменной $calc, и, затем, закончите блок PHP:   if ($calc == "сложить") {  $result = $val1 + $val2; } else if ($calc == "вычесть") {  $result = $val1 - $val2; } else if ($calc == "умножить") {  $result = $val1 \* $val2; } else if ($calc == "разделить") {  $result = $val1 / $val2; } ?>   * Начните блок HTML:   <html> <head> <title>Результат вычисления</title> </head> <body>   * Используя смешивание кодов PHP и  HTML, введите строку, которая выведет результат вычисления на экран:   <p>Результат вычисления равен: <? echo "$result"; ?></p>   * Завершите блок HTML:   </body> </html>   * Сохраните созданный файл на Web-сервере с именем calculate.php. * Вызовите в браузере файл calculate\_form.html. * Нажмите кнопку Вычислить, ничего не вводя в поля формы. Браузер перегрузит страницу, потому, что вы не ввели значения в три требуемых поля. * Попробуйте заполнить только одно из полей, посмотрите, что произошло. * Проверьте правильность выполнения всех операций. * Добавьте на страницу с результатом вычисления гиперссылку для возврата на форму ввода данных.   2. Задание   * Создайте сценарий для решения квадратного уравнения по введенным коэффициентам * Создайте сценарий для вычисления площади треугольника по трем введенным сторонам. Осуществите проверку на правильность введенных значений (длины сторон должны быть положительны) и существование треугольника с указанными длинами сторон. |