

## Практическая работа №4

### «PHP: циклы и массивы»

**Цель работы:** познакомиться с различными видами циклических конструкций в языке PHP, а также работой с массивами.

### ЗАДАНИЯ

#### А. Подготовка к работе:

1. Запустите менеджер виртуальных машин VMware Player на **реальном** компьютере.
2. В программе VMware Player запустите образ **виртуального** компьютера **WWW** с предустановленным ПО: Apache (HTTP-сервер), PHP (интерпретатор) и MySQL (СУБД).
3. В адресной строке браузера с реального компьютера откройте поочередно ссылки <http://www/>, <http://www/info.php> и <http://www/sqltest.php>. Убедитесь в работоспособности перечисленных выше программных компонентов виртуального веб-сервера.
4. В проводнике на реальном компьютере введите адрес [\\www](http://www), укажите по запросу имя пользователя – Администратор; пароль – 1234. Отобразится список общих папок веб-сервера, в т.ч. папка www – **корневая папка сайта** (локальный путь – C:\wamp\www).

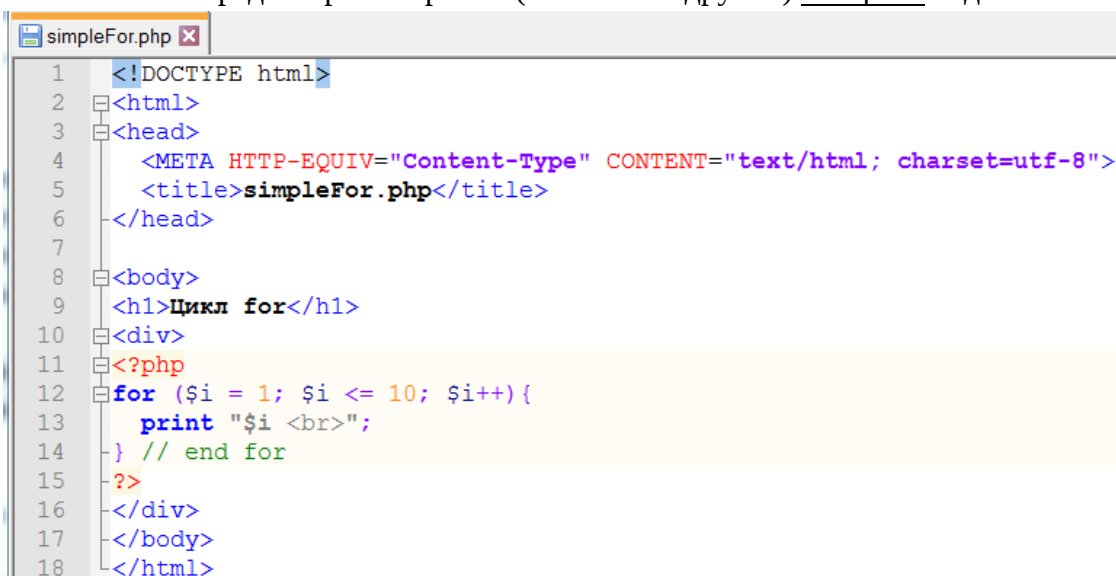
Если проводник сообщает, что нет доступа к ресурсу, то на реальном компьютере (для ОС Windows Vista / 7 / 8.x / 10) откройте «Панель управления» → «Сеть и Интернет» → «Центр управления сетями и общим доступом» → «Изменить дополнительные параметры общего доступа». В нужном профиле выберите «Включить сетевое обнаружение» и «Включить общий доступ, чтобы сетевые пользователи могли читать и записывать файлы в общих папках». Сохраните изменения и повторите попытку.

Возможной причиной блокирования также может быть **пустой пароль** у текущего пользователя реального компьютера.

*Примечание:* если запрещено изменение настроек на реальном компьютере, редактирование файлов сайта в дальнейшем возможно прямо на виртуальном компьютере.

#### Б. Цикл с параметром в PHP

5. В текстовом редакторе Notepad++ (или любом другом) наберите код:



```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=utf-8">
5      <title>simpleFor.php</title>
6  </head>
7
8  <body>
9      <h1>Цикл for</h1>
10     <div>
11         <?php
12         for ($i = 1; $i <= 10; $i++){
13             print "$i <br>";
14         } // end for
15         ?>
16     </div>
17 </body>
18 </html>

```

Сохраните код в файле simpleFor.php, проверьте его в браузере.

Команда **for** – это команда *цикла с параметром* в языке PHP. Здесь переменная `$i` – *параметр* с начальным значением 1. Повторяемое *тело цикла* (блок команд) – в фигурных скобках. В простых скобках через точку с запятой инициализируется параметр цикла, приводится *условие* выполнения тела цикла, а также механизм (способ) изменения значения параметра для следующей итерации (в нашем случае это инкремент, т.е. увеличение на 1). Значение, на которое изменяется параметр цикла – *шаг* – может быть любым (обычно целочисленным), в т.ч. и отрицательным.

Особенностью цикла с параметром как алгоритмической конструкции является то, что его используют, когда *заранее известно*, сколько раз должно повториться тело цикла. Цикл прекратится, когда параметр примет граничное значение.

Примечание: значение параметра может в теле цикла использоваться, но не должно в теле цикла изменяться, иначе как это предусмотрено механизмом при объявлении цикла.

В приведённом выше примере в цикле с параметром, принимающем значения от 1 до 10, с шагом +1 (т.е. 10 раз) повторяется вывод текущего значения параметра.

6. Измените код предыдущего примера так, чтобы:

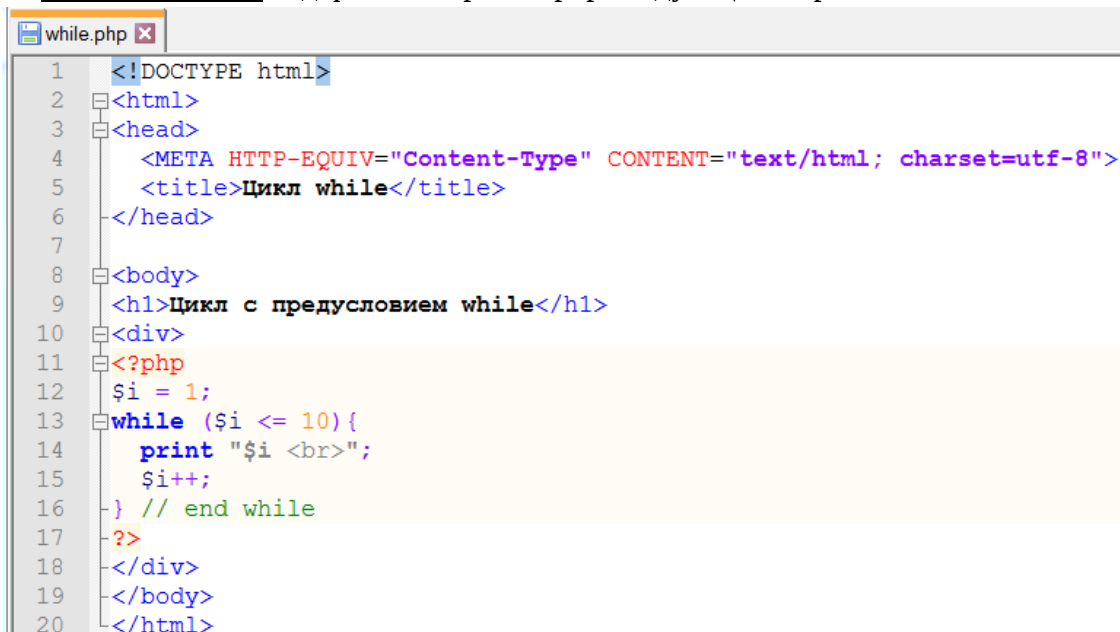
- а. вывод осуществлялся в обратном порядке – от 10 до 1;
- б. выводились значения от -25 до 25 с шагом +5.

В обоих случаях код сохраните в разных файлах.

7. Реализуйте скрипт, чтобы из заданного с помощью элементов формы диапазона лет был выведен на экран список високосных (определите для этого функцию, в которой используйте код задания 7 работы №3). Сопроводите ввод и вывод соответствующими комментариями для пользователя.

## В. Цикл while

8. Отредактируйте код файла simpleFor.php следующим образом:



```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=utf-8">
5      <title>Цикл while</title>
6  </head>
7
8  <body>
9      <h1>Цикл с предусловием while</h1>
10     <div>
11         <?php
12             $i = 1;
13             while ($i <= 10){
14                 print "$i <br>";
15                 $i++;
16             } // end while
17         ?>
18     </div>
19 </body>
20 </html>

```

Проверьте его в браузере.

Команда **while** определяет начало алгоритмической конструкции, называемой *цикл с предусловием*. В круглых скобках – *условие продолжения* цикла (очередной итерации), в

нашем примере в условии используется переменная `$i` (она должна быть инициализирована до начала цикла).

В фигурных скобках – повторяемое тело цикла (блок команд). В нашем примере здесь выводится текущее значение переменной `$i` и инкрементируется её значение для следующей итерации (или для дальнейшего использования получившегося значения после окончания цикла).

В отличие от цикла с параметром, цикл `while` используют, когда *заранее неизвестно* количество итераций – цикл будет продолжаться до наступления ложности условия, а когда это произойдет, программисту точно не известно.

**Примечание:** потенциально это чревато возможностью *бесконечного цикла* (заикливания).

Обычно причиной бесконечного цикла является опечатка в имени или ошибка в регистре. Задача программиста – сформировать код, который бы исключал возможность заикливания. Пример кода, порождающего бесконечный цикл:

```
$i = 1;
while ($i <= 10) {
    print "$i <br>";
    $j++;
} // end while
```

Заикливание скрипта может весьма серьезно снизить производительность веб-сервера. В большинстве случаев веб-сервер настроен так, чтобы остановить заикливание было возможным по нажатию в браузере кнопки остановки загрузки.

Может ли тело цикла `while` не выполниться ни разу?

9. Напишите код скрипта, определяющий, сколько различных цифр содержится во введенном натуральном числе.

## Г. Цикл `do-while`

Команда **`do-while`** определяет алгоритмическую конструкцию, называемую *цикл с постусловием*.

```
$i = 0;
do {
    print "$i <br>";
} while ($i > 0);
```

Здесь в круглых скобках – *условие продолжения* цикла (очередной итерации). В фигурных скобках – повторяемое тело цикла (блок команд).

Так же, как и цикл с предусловием, цикл `do-while` используют, когда *заранее неизвестно* количество итераций – цикл будет продолжаться до наступления ложности условия.

Однако, в отличие от цикла `while`, тело цикла `do-while` выполнится хотя бы один раз, т.к. проверка условия выхода осуществляется после выполнения очередной итерации.

## Д. Прерывание цикла

Для принудительного прерывания выполнения цикла язык PHP предоставляет команды `break` и `continue`.

Команда `break` позволяет прервать выполнение цикла до наступления условия и перейти к следующему за циклом выражению.

Команда `continue` позволяет прервать выполнение текущей итерации и начать выполнение новой.

Примечание: искусственные приемы прерывания и перезапуска цикла считается плохим стилем программирования, старайтесь не использовать эти команды.

10. Индивидуальное задание (задача №4) на циклы (см. по [1]). Снабдите вывод подробным описанием самого задания, переменных и результата. Реализуйте по возможности ввод данных через элементы формы.

## Е. Массивы

Массив в PHP – это составной тип данных, предназначенный для хранения списков как однотипных, так и разнотипных значений.

С примером массива вы уже сталкивались при реализации отправки данных из формы – массив \$\_GET для метода отправки GET и массив \$\_POST для метода POST.

Наиболее простой вид массива – индексный, к элементам которого можно обращаться по номерам.

11. Наберите следующий код:

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=utf-8">
5      <title>Массивы</title>
6  </head>
7
8  <body>
9      <h1>Примеры с массивами</h1>
10
11  <?php
12      //простой ввод данных в массив
13      $camelPop[1] = "Сомали";
14      $camelPop[2] = "Судан";
15      $camelPop[3] = "Мавритания";
16      $camelPop[4] = "Пакистан";
17      $camelPop[5] = "Индия";
18
19      //вывод данных из массива в цикле
20      print "<h3>Страны мира с наибольшим поголовьем верблюдов:</h3>";
21      print "<p> ";
22      for ($i = 1; $i <= 5; $i++){
23          print "$i: $camelPop[$i]<br>";
24      } // end for
25      print "</p>";
26
27      //функция array для задания значений массива
28      $binary = array("000", "001", "010", "011");
29
30      //снова вывод данных из массива
31      print "<h3>Двоичные числа:</h3>";
32      print "<p> ";
33      for ($i = 0; $i < count($binary); $i++){
34          print "$i: $binary[$i]<br>";
35      } // end for
36      print "</p>";
37  ?>
38  </body>
39  </html>

```

Проверьте работу скрипта в браузере.

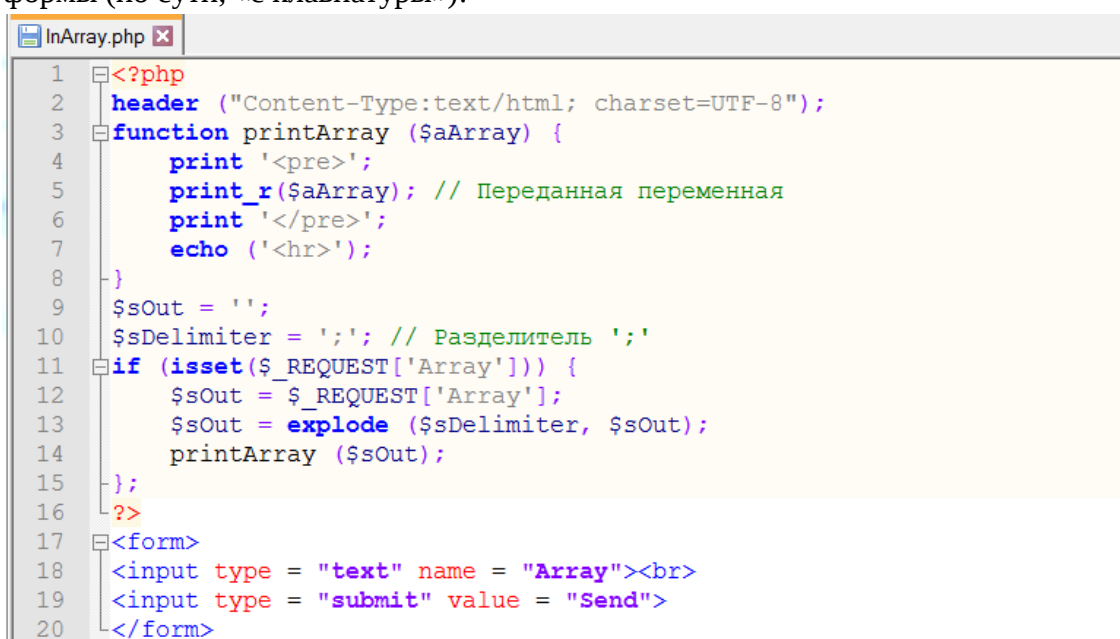
В этом примере элементы первого массива \$camelPor заданы операторами присваивания. Элементы второго массива \$binary – с помощью функции array(). Вывод значений в обоих случаях осуществлялся в цикле по номерам (индексам) элементов этих массивов.

Примечание: обратите внимание – для массива \$camelPor явно задано, что номер его первого элемента – 1. Функция array() первому элементу массива \$binary автоматически присвоила индекс 0.

При выводе второго массива использована функция count(), возвращающая длину (количество элементов) массива. Представьте этот код вывода в виде функции, которую впоследствии можно будет использовать в других задачах на массивы.

12. Напишите и проверьте в браузере код функции для задания целочисленных массивов случайными числами в диапазоне от -200 до 200.

13. Наберите и проверьте работу кода, позволяющего ввести значения массива из элемента формы (по сути, «с клавиатуры»):



```

1  <?php
2  header ("Content-Type:text/html; charset=UTF-8");
3  function printArray ($aArray) {
4      print '<pre>';
5      print_r($aArray); // Переданная переменная
6      print '</pre>';
7      echo ('<hr>');
8  }
9  $sOut = '';
10 $sDelimiter = ';'; // Разделитель ';'
11 if (isset($_REQUEST['Array'])) {
12     $sOut = $_REQUEST['Array'];
13     $sOut = explode ($sDelimiter, $sOut);
14     printArray ($sOut);
15 };
16 ?>
17 <form>
18 <input type = "text" name = "Array"><br>
19 <input type = "submit" value = "Send">
20 </form>
  
```

С помощью справочника по PHP изучите все использованные здесь встроенные функции. Сделайте из представленного кода функцию (в дальнейшем используйте её в других задачах на массивы).

14. Напишите скрипт для сортировки целочисленного массива методом пузырька. Значения элементов массива задаются генератором случайных чисел. При выводе результата для наглядности отображаются исходный и отсортированный массивы.

15. Индивидуальное задание (задача №5) на массивы (см. по [1]). Снабдите ввод и вывод подробным описанием самого задания, переменных и результата.

#### Литература:

1. Абрамов С.А. и др. Задачи по программированию, 1988.
2. Маклафлин Б. PHP и MySQL. Исчерпывающее руководство, 2013.
3. Суэринг С. и др. PHP 6 и MySQL 6. Библия программиста, 2010.
4. Янк К. PHP и MySQL. От новичка к профессионалу, 2013.
5. PHP: Справочник языка – Manual [Электронный ресурс]. URL: <http://php.net/manual/ru/langref.php> (дата обращения 25.03.2016).