Лабораторная работа №3.

Работа с массивами данных и файлами в PHP

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Работа с массивами данных

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

Массив можно создать двумя способами:

1. С помощью конструкции array

```
$array_name = array("key1"=>"value1", "key2"=>"value2");
```

2. Непосредственно задавая значения элементам массива

```
$array_name["key1"] = value1;
```

Например, нам нужно хранить список документов, которые будут удалены из базы данных. Естественно хранить его в виде массива, ключом в котором будет идентификатор документа (его уникальный номер), а значением – название документа. Этот массив можно создать таким образом:

```
<?php
  $del_items = array("10"=>"Наука и жизнь", "12"=>"Информатика");
  // добавляем элемент в массив
  $del_items["13"] = "Программирование на Php";
?>
```

Объединение массивов выполняют с помощью стандартного оператора «+».Если имеется два массива, $a и $b, то результатом их 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.

```
<?php
  $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)
?>
```

Сравнивать массивы можно, проверяя их равенство или неравенство либо эквивалентность или неэквивалентность. Равенство массивов – это когда совпадают все пары ключ/значение элементов массивов. Эквивалентность – когда кроме равенства значений и ключей элементов требуется еще, чтобы элементы в обоих массивах были записаны в одном и том же порядке. Равенство значений в PHP обозначается символом «= =», а эквивалентность – символом «= = =».

```
<?php
  $a = array("и"=>"Информатика", "м"=>"Математика");
  $b = array("м"=>"Математика", "и"=>"Информатика");
  if ($a == $b) echo "Массивы равны и";
  else echo "Массивы НЕ равны и ";
  if ($a === $b) echo " эквивалентны";
  else echo " НЕ эквивалентны";
  // получим echo "Массивы равны и НЕ эквивалентны"
?>
```

Далее рассмотрим еще одну важную операцию с массивом – подсчет количества его элементов. Для ее реализации в PHP есть специальная функция.

Функция count() вычисляет число элементов в переменной вообще. Если применить ее к любой другой переменной, она возвратит 1. Исключение составляет переменная типа NULL – count(NULL) есть 0. Кроме того, применяя эту функцию к многомерному массиву, чтобы получить число его элементов, нужно использовать дополнительный параметр COUNT_RECURSIVE.

```
<?php
 $del_items = array("langs" => array("10"=>"Python", "12"=>"Lisp"), "other"=>"Информатика");
 echo count($del_items)."<br>";
 // выведет 2
 echo count($del_items, COUNT_RECURSIVE);
 // выведет 4
?>
```

Функция in_array позволяет установить, содержится ли в заданном массиве искомое значение.

```
in_array("искомое значение", "массив", ["ограничение на тип"]);
```

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.

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.

Функция array_keys() выбирает все ключи массива. Но у нее имеется дополнительный аргумент, с помощью которого можно получить список ключей элементов с конкретным значением. Синтаксис этой функции таков:

```
array_keys ("массив", ["значение для поиска"]);
```

Функция array_keys() возвращает как строковые, так и числовые ключи массива, организуя все значения в виде нового массива с числовыми индексами.

Функция sort сортирует массив и имеет следующий синтаксис

sort (массив [, флаги]);

т.е. упорядочивает его значения по возрастанию. Эта функция удаляет все существовавшие в массиве ключи, заменяя их числовыми индексами, соответствующими новому порядку элементов. В случае успешного завершения работы она возвращает true, иначе – false.

В качестве дополнительного аргумента флаги может использоваться одна из следующих констант:

- SORT_REGULAR – сравнивать элементы массива обычным образом;
- SORT_NUMERIC – сравнивать элементы массива как числа;
- SORT_STRING – сравнивать элементы массива как строки.

Если требуется сохранять индексы элементов массива после сортировки, то нужно использовать функцию asort (массив [, флаги]). Если необходимо отсортировать массив в обратном порядке, т.е. от наибольшего значения к наименьшему, то можно задействовать функцию rsort (массив [, флаги]). А если при этом нужно еще и сохранить значения ключей, то следует использовать функцию arsort(массив [, флаги]).

Функция array_slice выделяет подмассив длины длина в массиве массив, начиная с элемента, номер которого задан параметром номер_элемента. Синтаксис:

array_slice (массив, номер_элемента [, длина]);

Функция array_sum() вычисляет сумму всех элементов массива.

Работа с файлами

В PHP не существует функции, предназначенной именно для создания файлов. Большинство функций работают с уже существующими файлами в файловой системе сервера. Для того, чтобы создать самый обычный файл, нужно воспользоваться функцией, которая открывает локальный или удаленный файл - это функция fopen(). Открыть файл означает, что fopen связывает данный файл с потоком управления программы. Причем связывание бывает различным в зависимости от того, что мы хотим делать с

этим файлом: читать его, записывать в него данные или делать и то и другое. Синтаксис:

resource fopen (имя_файла, тип_доступа [, use_include_path]);

В результате работы эта функция возвращает указатель (типа ресурс) на открытый ею файл. В качестве параметров этой функции передаются: имя файла, который нужно открыть, тип доступа к файлу (определяется тем, что мы собираемся делать с ним) и, возможно, параметр, определяющий, искать ли указанный файл в include_path.

Параметр имя_файла должен быть строкой, содержащей правильное локальное имя файла или URL-адрес файла в сети. Если имя файла начинается с указания протокола доступа (например, http://... или ftp://...), то интерпретатор считает это имя адресом URL и ищет обработчик указанного в URL протокола. Если обработчик найден, то PHP проверяет, разрешено ли работать с объектами URL как с обычными файлами (директива allow_url_fopen). Если allow_url_fopen=off, то функция fopen вызывает ошибку и генерируется предупреждение. Если имя файла не начинается с протокола, то считается, что указано имя локального файла. Чтобы открыть локальный файл, нужно, чтобы PHP имел соответствующие права доступа к этому файлу.

Параметр use_include_path, установленный в значение 1 или TRUE, заставляет интерпретатор искать указанный в fopen() файл в include_path. Напомним, что include_path - это директива из файла настроек PHP, задающая список директорий, в которых могут находиться файлы для включения. Кроме функции fopen() она используется функциями include() и require().

Параметр тип_доступа может принимать одно из следующих значений.

Тип доступа	**Описание**
r	Открывает файл только для чтения; устанавливает указатель позиции в файле на начало файла.
r+	Открывает файл для чтения и записи; устанавливает указатель файла на его начало.
w	Открывает файл только для записи; устанавливает указатель файла на его начало и усекает файл до нулевой длины. Если файл не существует, то пытается

	создать его.
w+	Открывает файл для чтения и записи; устанавливает указатель файла на его начало и усекает файл до нулевой длины. Если файл не существует, то пытается создать его.
a	Открывает файл только для записи; устанавливает указатель файла в его конец. Если файл не существует, то пытается создать его.
a+	Открывает файл для чтения и записи; устанавливает указатель файла в его конец. Если файл не существует, то пытается создать его.
x	Создает и открывает файл только для записи; помещает указатель файла на его начало. Если файл уже существует, то fopen() возвращает false и генерируется предупреждение. Если файл не существует, то делается попытка создать его. Этот тип доступа поддерживается начиная с версии PHP 4.3.2 и работает только с локальными файлами.
x+	Создает и открывает файл для чтения и записи; помещает указатель файла на его начало. Если файл уже существует, то fopen() возвращает false и генерируется предупреждение. Если файл не существует, то делается попытка создать его. Этот тип доступа поддерживается, начиная с версии PHP 4.3.2, и работает только с локальными файлами.

Если открыть или создать файл с помощью fopen не удается, PHP генерирует предупреждение, а функция fopen возвращает как результат своей работы значение false. Такого рода предупреждения можно «подавить» (запретить) с помощью символа @.....

После выполнения необходимых действий с файлом, будь то чтение или запись данных или что-либо другое, соединение, установленное с этим файлом функцией fopen(), нужно закрыть. Для этого используют функцию fclose(). Синтаксис у нее следующий:

fclose (указатель на файл);

Эта функция возвращает TRUE, если соединение успешно закрыто, и FALSE - в противном случае.

Для того чтобы удалить файл с помощью языка PHP, нужно воспользоваться функцией unlink(). Синтаксис этой функции можно описать следующим образом:

bool unlink (имя_файла);

Данная функция удаляет файл, имеющий имя имя_файла, возвращает TRUE в случае успеха этой операции и FALSE - в случае ошибки. Чтобы удалить файл, нужно тоже иметь соответствующие права доступа к нему

ПРАКТИЧЕСКАЯ ЧАСТЬ

1. Создайте в папке mylabs.ru файл с именем index.php и следующим содержанием:

```
<html>
<head>
<title> Работа с массивами и файлами</title>
</head>
<body bgcolor=yellow>
<?PHP
$url = "mas.txt";
if (isset($_POST[element])) {
    if(!file_exists($url)) {
        $handle = fopen($url,"a");
        $str=$_POST[element];
    } else {
        $handle = fopen($url,"a+");
        $str="\n".$_POST[element];
    }
    fputs ($handle,$str,20);
    fclose($handle);       //закрываем файл
    $arr=file($url);  //Чтение данных из файла в массив
    echo "Введите ";
    echo count($arr);
}
else {
    if(file_exists($url))  {
        $arr=file($url);                              //Чтение данных из файла в массив
        echo "Введите ";
        echo count($arr);
    } else {
        echo "Введите 0";
    }
}
echo "-й элемент массива";
print( "<form action=$_SERVER[PHP_SELF] method=POST>");
print("<input type=text  name=\"element\"> <input type=submit name=\"submit\"
value=\"Добавить\">");
print("<table border=1> <tr><td colspan=2> Содержание массива</td>");
print("<tr> <th>№</th> <th>Элемент массива</th>");
```

```
for ($x=0;$x<count($arr);$x++) //Вывод массива на экран в виде таблицы
{ echo "<tr> <th>".$x."</th>  <th>".$arr[$x]."</th>"; }
print("</table>");
print("</form>");
?>
</body>
</html>
```

Здесь PHP_SELF в строке

```
print( "<form action=$_SERVER[PHP_SELF] method=POST>");
```

это встроенная переменная окружения, хранящая имя отображаемой страницы.

2. Пример скрипта реализующего счетчик посещаемости страницы ВАРИАНТ 1.

Чтобы использовать счетчик, нам необходимо создать html-страницу, в которую будет встроен скрипт. Пусть это будет простая страница, которая содержит информацию.

index.php

```
<html>
<head>
<title>Моя страничка</title>
</head>
<body bgcolor="white" text="black" link="blue" vlink="purple" alink="red">
<p>Добро пожаловать</p>
</body>
</html>
```

Сохраним эту страницу в папке www и перейдем к программированию счетчика.

counter.php

Чтобы сохранять полученные данные, нам необходимо их куда то записывать, пусть для начала это будет простой текстовый файл с именем stat.txt.

```
<?php
$url = "stat.txt";
```

Переменной $url присвоено имя файла.

Далее идет фрагмент кода, который отвечает за проверку на наличие файла.

```
if(!file_exists($url)) {                    //Проверка на существование файла.
//Если файла не существует,
$count = 0;
$handle = fopen($url,"a");                  //создаем его,
fwrite($handle,$count);                     // и записываем нулевое значение
fclose($handle);                            //Закрываем файл
}
```

Далее опишем условие, когда файл создан.

```
else {                                      //Если файл существует, то работаем с ним
$handle = fopen($url,"a+");                 //Открываем его
$count = fread($handle,filesize($url));  //Читаем данные в переменную $count
fclose($handle);                            //Закрываем файл
$count++;                                 //Добавляем одно посещение к полученному выше
$handle = fopen($url,"w"); // Открываем файл, и урезаем его до нулевой длины
fwrite($handle,$count);     //Записываем переменную $count
fclose($handle);                          //Закрываем файл
}
```

В этом фрагменте описан **скрипт счетчика посещений.** Сначала мы открываем файл для того чтобы узнать количество посещений. Далее увеличиваем полученное значение на 1 единицу и записываем это значение обратно в файл.

В конце скрипта выводим статистику при помощи **оператора echo**

```
echo "количество просмотров: $count ";?>
```

Далее чтобы этот счетчик заработал добавим в файл index.php следующую строчку перед тэгом </body>

```
<?php include("counter.php"); ?>
```

Проверьте работу полученного счетчика.

Основным недостатком данного счетчика является, то что он легко накручивается простым обновлением страницы. также если несколько человек одновременно обратятся к скрипту, возникнет ошибка, так как файл открыт и используется кем то другим.

ВАРИАНТ 2.

В начале файла index.php добавьте следующую строку

```
<?php session_start();   ?>
```

Функция session_start() инициализирует сессию.

Измените содержимое файла counter.php

```
<?php
$url = "stat.txt";
if(!file_exists($url)) {
    $count = 0;
    $handle = fopen($url,"a");
    fwrite($handle,$count);
    fclose($handle);
}
else {
    $handle = fopen($url,"a+");
    $count = fread($handle,filesize($url));
```

```
    fclose($handle);
    if (!isset($_SESSION['cnt'])) {        //Если сессия отсутствует
        $_SESSION['cnt'] = 1;              //Создаем её
        $count++;                          //Добавляем единицу к переменной
        $handle = fopen($url,"w");
        if (flock($handle, LOCK_EX)) {
            fwrite($handle,$count);        //Записываем её в файл
             flock($handle, LOCK_UN);      // отпираем файл
        } else {
            echo "Повторите позднее";
        }
        fclose($handle);
    }
}
//Выводим количество просмотров.
 echo "количество просмотров: $count";
?>
```

Здесь добавлено условие, если сессия отсутствует, то это значит человек посещает страницу в первый раз, поэтому переменной $_SESSION['cnt'] присваиваем какое то значение, которое проверяется в начале условия, далее к переменной $count добавляем единицу, и записываем её обратно в файл, причем для начала запираем доступ к нему, производим запись и открываем доступ. Это нужно для того чтобы в момент обращения нескольких посетителей к скрипту не возникало ошибок.

В данном примере при обновлении страницы счетчик посещений не увеличивается, так как при первом посещении мы создаем сессию, то в результате проверки if(!isset($_SESSION['cnt']) выражение вернет значение **FALSE**, следовательно ничего прибавлять и записывать не будем.

ЗАДАНИЕ

1. Напишите скрипт позволяющий ввести массив чисел и вывести его элементы в порядке возрастания.

2. Организуйте ввод двумерного массива.

3. Дан непустой массив из цифр. Вывести на экран цифру, наиболее часто встречающуюся в этом массиве.

4. Преобразовать массив X по следующему правилу: все отрицательные элементы массива перенести в начало, а все остальные – в конец, сохраняя исходное взаимное расположение, как среди отрицательных, так и среди остальных элементов.

5. Задан массив. определить упорядочен ли данный массив по убыванию.

6. Элементы каждого из массивов X и Y упорядочены по неубыванию. Объединить элементы этих двух массивов в один массив Z так, чтобы они снова оказались упоряченными по неубыванию.

7. Дан массив из k символов. Определить, симметричен ли он, т.е. читается ли он одинаково слева направо и справа налево.

8. Дано два массива. Найти наименьшее среди тех элементов первого массива, которые не входят во второй массив.

9. Вывести на экран заданный массив из k символов, удалив из него повторные вхождения каждого символа.

10. Определить сколько различных символов входит в заданный массив.

11. Напишите скрипт позволяющий ввести массив чисел и вывести его элементы в порядке убывания.

12. Дан непустой массив из цифр. Вывести на экран цифру, наиболее часто встречающуюся в этом массиве.

13. Задан массив. Определить k – количество “особых” элементов массива, считая элемент “особым”, если он больше суммы остальных элементов.

14. Преобразовать массив X по следующему правилу: все отрицательные элементы массива перенести в конец, а все остальные – в начало, сохраняя исходное взаимное расположение, как среди отрицательных, так и среди остальных элементов.

15. Элементы каждого из массивов X и Y упорядочены по убыванию. Объединить элементы этих двух массивов в один массив Z так, чтобы они снова оказались упорядоченными по убыванию.

16. Дан массив из k символов. Определить, симметричен ли он, т.е. читается ли он одинаково слева направо и справа налево.

17. Дано два массива. Найти наименьшее среди тех элементов первого массива, которые не входят во второй массив.

18. Вывести на экран заданный массив из k символов, удалив из него повторные вхождения каждого символа.

19. Определить сколько различных числе входит в заданный массив.

20. Задан массив. Определить k – количество “особых” элементов массива, считая элемент “особым”, если слева от него находятся элементы, меньшие его, а справа – б*о*льшие.

21. Задан массив. определить упорядочен ли данный массив по возрастанию.

22. Вывести на экран заданный массив из k символов, удалив из него повторные вхождения каждого символа.

23. Дан непустой массив из цифр. Вывести на экран цифру, наиболее часто встречающуюся в этом массиве.

24. Преобразовать массив X по следующему правилу: все отрицательные элементы массива перенести в начало, а все остальные – в конец.

25. Задан массив. определить упорядочен ли данный массив по убыванию или по возрастанию.