PHP

Лектор:
Калюжняк А.В.

- PHP – це гнучкий і легкий мову web-програмування, що має широкі можливості та незаперечні переваги.
- PHP це мова програмування, за допомогою якої створюють сайти, що активно взаємодіють з користувачем, наприклад: Інтернет-магазин, веб-каталоги, поштову розсилку на сайті, стрічку новин, довідники, форуми та багато іншого.
- PHP відрізняється від інших подібних мов (JavaScript) тим, що код виконується на сервері. Якщо ви маєте скрипт на сервері, клієнт отримає результат роботи цього скрипту, не маючи можливості визначити, який був вихідний код.
- Найкращою якістю PHP є те, що він простий для новачка в програмуванні і пропонує багато можливостей для програміста-професіонала.

- **1994 - програміст Расмус Лердорф написав власні Perl-скрипти для власної сторінки і назвав Personal Home Page (PHP)**

Програмування PHP. Історія розвитку.

1997 - два програмісти Енді Гутманс і Зів Сураскі взяли за основу ідею Расмуса і переписали PHP з нуля. Назва "Personal Home Page " было изменено на **Hypertext Preprocessor**

- 2000 рік – вийшла 4-та версія PHP, що стала стандартом для веб-розробки. Зараз розроблено PHP5.
- PHP – це гнучкий і легкий мову web-програмування, що має широкі можливості та незаперечні переваги.

Сьогодні PHP - це потужний кросплатформовий набір засобів, що розташовується на сервері і призначений для обробки спеціального коду, що вбудовується в HTML-сторінку. Завдяки цьому з'являється можливість легко створювати динамічні сайти. Файли, створені таким чином, зберігаються та обробляються на сервері, і коли відвідувач запитує документ із PHP, скрипт обробляється не браузером відвідувача, як, наприклад, Java Script, а сервером, і відвідувачу передаються вже лише результати роботи.

Динамічний сайт, як правило, повністю управляється через нескладний веб-інтерфейс. Управління можливе не лише окремими сторінками, а й структурою розділів та інформаційною сіткою сайту.

Управління сайтом доступне (і рекомендується) менеджерам, які безпосередньо спілкуються з клієнтами та знають які питання потрібно оперативно висвітлити на сайті.

Важливий плюс – це оперативність публікації нових матеріалів, оголошень та іншої важливої інформації, що робить спілкування з відвідувачами (клієнтами) сайту «живим» та цікавим.

Основні способи вставки коду PHP

```
<?php
   // Тут будет Php код
?>
```

PHP код складається з інструкцій, розділених знаком;

Правильні записи

```
<?php
   інструкція1;
   інструкція2;
?>
```

```
<?php
   інструкція1;  інструкція2;
?>
```

Виведення тексту на екран. Оператор Echo.

Коли потрібно відобразити текст на веб-сторінці, то оператор echo є найбільш уживаним оператором в PHP. Як його використовувати - після слова echo потрібно помістити рядок тексту в лапки:

```
<?php
    echo 'Привіт від PHP';
        ?>
```

Відображення у браузері:
Привіт від PHP
(Для відображення тексту можна використовувати як подвійні лапки, так і одинарні).

Для чисел лапки можна не використовувати:

```
<?php
  echo 2023;
  ?>
```

Коментарі

Однострочные

```
<?php
    // Это комментарий
    # Это тоже комментарий
?>
```

```
<?php
    /* Это комментарий
    многострочный
    echo "Привет";
    */
?>
```

Змінні в PHP

PHP створено не лише для форматування статичного тексту. Для того, щоб обробляти різні дані, були придумані змінні.

Кожна змінна містить певне значення.

Синтаксис змінної складається із знака долара -$ і "вільного" ідентифікатора якому присвоюється якесь значення. Наприклад:

```
<?php $name = "Віктор"; ?>
```

Створення змінної

Змінна створюється тоді, коли їй надають якесь значення. Для визначення значення змінної використовують оператор присвоєння. Наприклад:

```
<?php
$surname = "Петров";
$number = 1269794645;
$pi = 3.14159265;
$hello = "Hi all";
?>
```

Змінну можна вивести на екран за допомогою оператора echo, ось так:

```
<?php
$name = "Віктор";
echo "Ваше імя ", $name, "<br>";
?>
```

Відображення у браузері:Ваше імя Віктор

Створимо змінну яка міститиме значення кількості бананів, друга змінна кількість лимонів, а третя - їх сумарна кількість.

```
<?php
$bann = 5;   // Банани
$lim = 10;    // Лимони
$together = $bann + $lim; // Всього
echo "Кількість фруктів ", $together;
?>
```

Відображення у браузері:

Кількість фруктів 15

Інтерполяція змінних у PHP

Значення змінної може бути відображено наприклад так :

```
<?php
$capital = "Paris";
echo "The capital of France is", $capital,"<br />";
?>
```

Але є спосіб зробити це найпростіше. Якщо ім'я змінної укладено в подвійні (не одинарні) лапки, змінна інтерполується. Наприклад:

```
<?php
$capital = "Paris";
echo "The capital of France is $capital <br />";
?>
```

Відображення у браузері:

The capital of France is Paris

Змінні, що містять імена інших змінних

У PHP можна розміщувати значення змінних як звичайні значення, а й імена інших змінних.

```
<?php
$apples = 5;
$fruit = "apples"; /* Створюємо змінну $fruit, що містить
                      ім'я змінної $apples */
// Зараз ми можемо вивести $apples, як $$fruit
echo "Число яблук - ", $$fruit; ?>
```

Для коректного відображення подібних змінних у рядкових константах, укладених у подвійні лапки, слід також використовувати фігурні дужки:
${$fruit}.
Наприклад:

```
<?php
echo "Число яблук - ${$fruit}";
?>Число яблук - 5
```

Константи в PHP

Коли не потрібно змінювати задане значення для змінної, то є сенс створити константу і потім використовувати її в будь-якій частині скрипта. Для опису константи використовують функцію define, якій передається її ім'я та значення:

```
<?php
    define("pi", 3.14); ?>
```

Ім'я константи потрібно завжди укладати в лапки, а її значення лише тоді, коли воно є рядком.

Приклад використання константи:

```
<?php
    define("pi", 3.14);
    echo " Математична константа Пі дорівнює ", pi; ?>
```

Відображення у браузері:

Математична константа Пі дорівнює 3.14

Спроба зміни константи призведе до непрацездатності скрипту.

Типи даних PHP

PHP є мовою динамічної типізації (тип змінної визначається на основі її значення).

Типи, які можна використовувати в PHP:

Boolean. Це логічний тип, який містить значення TRUE чи FALSE.

Integer. Містить значення цілого числа (наприклад: 4 або 10 або інше ціле число).

String. Містить значення тексту довільної довжини (наприклад, Олег, Київ, Австрія).

Float. Речовище (Наприклад: 1.2, 3.14, 8.5498777).

Object. Об'єкт.

Array. Масив.

Resource. Ресурс (наприклад: файл).

NULL. Значення NULL.

```php
<?php
$bool = TRUE; // Значення Boolean
$int = 100; // Значення Integer
$string = "Переменная содержит текст"; // Значення String
$string2 = "5425"; // Значення String, так как число взято в кавычки
! $float = 44.122; // Значення Float
?>
```

Для запобігання появі помилок рекомендується не змішувати різні типи даних. Якщо треба змінити тип змінної даних, то для цього потрібно зліва від імені змінної в круглих дужках вказати потрібний тип:

```php
<?php
$str = "50000"; // Значення String
$new_str = (integer) $str; // Тепер значення стало Integer
echo $new_str + $new_str;
?>
```

Відображення у браузері:100000

Математичні оператори та математичні функції PHP

Числові дані обробляються за допомогою таких операторів:
+, -, *, /, % (залишок від розподілу)

```
<?php
    echo "2 + 2 = ", 2 + 2, "<br>";
    echo "5 - 2 = ", 5 - 2, "<br>";
    echo "10 * 10 = ", 10 * 10, "<br>";
    echo "100 / 2 = ", 100 / 2, "<br>";
    echo "10 % 2 = ", 10 % 2, "<br>";
    ?>
```

Відображення в браузері

2 + 2 = 4
5 - 2 = 3
10 * 10 = 100
100 / 2 = 50
10 % 2 = 0

```php
<?php
echo "round(4.2) = ", round(4.2), "<br>";
?>
```

Відображення в браузері

round(4.2) = 4

Оператори присвоєння в PHP

Основним оператором присвоєння є знак рівності =. Він надає значення певної змінної:

```
<?php $fruits=14; ?>
```

В одному рядку можна надати одне значення відразу декільком змінним, наприклад:

```
<?php $n = $m = $p = 3; echo $n, $m, $p; ?>
```

Відображення в браузері:

333

Також у PHP є комбіновані оператори, які роблять код компактнішим. Ось їх перелік:

+=, -=, /=, .=, %=, &=, |=, ^=, <=, >=

Наприклад, якщо потрібно додати 55 до значення змінної $number, це можна записати як: $number = $number + 55, а якщо використовувати комбінований оператор, то так:

$number += 55.

Збільшення та зменшення на 1

Якщо є змінна $a = 0, то щоб додати 1 до цієї змінної потрібно написати: $a++, якщо потрібно відібрати 1, то потрібно записати так: $a--. Оператор ++ називають інкрементом, а -- декрементом.

Оператор виконання PHP

У PHP існує такий оператор, як оператор виконання, він потрібен для того щоб виконувати команди ОС і використовувати результат цього виконання.

Будь-який рядок, який укладений у зворотні апострофи — вважаються як команда ОС. Наприклад:

```
<?php
$d = `dir d:\\`;
echo $d;
?>
```

Як результат, ви отримаєте список директорій диска D.

Рядкові оператори PHP

PHP має два рядкові оператори.

Перший - оператор конкатенації, який об'єднує два рядки в один.

Другий - конкатенуючий оператор присвоєння. =, додає до рядка потрібне значення. Наприклад:

```
<?php
$d = "Hello";
$f = $d." world"; // Тепер $f = "Hello world"
echo $f;
echo "<br/>";
$f .= " ! ! !"; // Тепер $f = "Hello world ! ! !"
echo $f;
?>
```

Відображення у браузері :

Hello world

Hello world ! ! !

Умовний оператор IF в PHP

У всіх високорівневих мовах програмування є оператор if, в PHP синтаксис цього оператора такий:

if (exp) statement

exp (вираз) - логічний вираз, який може бути істиною (TRUE) або брехнею (FALSE).

statement (інструкція) виконується тоді, коли exp — істина, і не виконується коли exp;

Наприклад, якщо швидкість машини буде більше 60, то це означає, що водій перевищує швидкість:

```
<?php
        $speed = 80;
        if ($speed > 60)
            echo "Перевищення швидкості !";
        ?>
```

Якщо потрібно, щоб при виконанні умови виконувались відразу кілька операторів, то потрібно укласти їх у фігурні дужки { }:

```
<?php
$speed = 80;
if ($speed > 60)
    {echo "Перевищення швидкості! <br>";
     echo "Будь ласка, зменшіть швидкість!"; }
?>
```

Відображення у браузері:

Перевищення швидкості!

Будь ласка, зменшіть швидкість!

Оператори порівняння PHP

Усі оператори порівняння PHP вказані в таблиці:

==	Рівність	Истина, если $a равно $b
===	Ідентичність	Истина, если $a равно $b, и они одного и того же типа
!=	Нерівність	Истина, если $a не равно $b
<>	Нерівність	Истина, если $a не равно $b
!==	НеІдентичність	Истина, если $a не равно $b, или они не одного типа
<	Меньше	Истина, если $a меньше $b
>	Більше	Истина, если $a больше $b
<=	Меньше або дорівнює	Истина, если $a меньше или равно $b
>=	Більше або дорівнює	Истина, если $a больше или равно $b

```php
<?php
$speed = 45;
if ($speed != 60)
      echo «Швидкість в межах норми";
?>
```

Якщо потрібно застосувати до висловлювання кілька умов, використовують логічні оператори:

```php
<?php
$speed = 40;
if ($speed > 35 && $speed < 55)
   { echo " Швидкість в межах норми "; }
?>
```

&&	Логічне «І"	Истина, если истинно $a и $b
||	Логічне «АБО"	Истина, если истинно $a или $b
xor	Логічне «Виключаюче АБО"	Истина, если истинно $a или $b, но не оба одновременно
!	Логічне "Ні"	Истина, если $a ложь

Оператор ELSE в PHP

Синтаксис оператора:

if(exp) statement1 **else** statement2

Приклад:

```
<?php
$speed = 50;
if ($speed > 60)
   echo "Перевищення швидкості !";
else
   echo «Швидкість в межах норми"
?>
```

У цьому випадку буде виведено повідомлення

Швидкість у межах норми

Оператор ELSEIF в PHP

Оператор if має ще одне розширення, це оператор elseif, він використовується для послідовної перевірки умов.

Синтаксис:

```
if (exp)
        statement1
    elseif (exp2)
        statement2
```

Також можна записувати так:

```
if (exp)
        statement1
    else if (exp2)
        statement2
```

Приклад:

```
<?php $speed = 50;
if ($speed < 30)
   echo " Швидкість в межах норми ";
elseif ($speed == 30)
   echo "Ваша швидкість 30 км/час";
elseif ($speed == 40)
   echo "Ваша швидкість 40 км/час";
elseif ($speed == 50)
   echo "Ваша швидкість 50 км/час";
elseif ($speed == 60)
   echo "Ваша швидкість 60 км/час";
else echo "Перевищення швидкості !"; ?>
```

Також такий шматок коду можна записати і так:

```
<?php
$speed = 50;
if ($speed < 30)
      echo " Швидкість в межах норми ";
elseif ($speed >= 30 && $speed <= 60)
       echo "Ваша швидкість {$speed} км/год";
else echo «Перевищення швидкості !";
?>
```

У цьому випадку буде виведено повідомлення **Ваша швидкість 50 км/год**. А якби не одна умова не підійшла б, то спрацював би оператор і ми побачили "**Перевищення швидкості!".**

Тернарний оператор PHP

Тернарний оператор працює майже як і оператор if, але при використанні тернарного оператора, ми замість ключових слів пишемо ? та:.

Синтаксис:

```
$var = condition ? exp1 : exp2;
```

Якщо умова виконується, то змінної $ var присвоюється результат обчислення exp1, інакше exp2.

Приклад:

```
<?php
$speed = 55;
echo ($speed <= 60) ? " Швидкість в межах норми " : "
Перевищення швидкості!"; ?>
```

В результаті ми побачимо рядок - "Швидкість у межах норми".

Оператор switch

Іноді використання конструкції операторів if .. elseif дещо втомлює. Щоб виправити цю ситуацію є оператор switch.

Синтаксис:

```
switch (exp)
        { case condition1: exp1; break;
        case condition2: exp2; break;
        case condition3: exp3; break;
        default: exp4;
        break; }
```

Спочатку записується ключове слово switch, після якого в дужках записується деякий вираз. Далі, після слова case потрібно перерахувати можливі варіанти значень, якщо значення істина, то виконується група операторів, які записані до оператора break. Якщо жодна умова не походить, то виконується оператор default (якщо оператор default не записувати, то при невиконанні жодних інших умов нічого не станеться).

```php
<?php
$speed = 55;
switch($speed)
{  case 30 : echo "Ваша швидкість 30 км/час"; break;
   case 58 : echo "Ваша швидкість 50 км/час"; break;
   case 70 : echo "Перевищення швидкості !"; break;
   default : echo " Швидкість в межах норми "; break; }
?>
```

Також, при використанні оператора **switch**, ми можемо записати кілька умов для деякої дії:

```php
<?php
$speed = 55;
switch($speed)
{ case 30 : case 58 : echo " Швидкість в межах норми "; break;
  case 70 : echo " Перевищення швидкості!"; break;
  default : echo " Швидкість в межах норми "; break; } ?>
```

В результаті ми побачимо — " Швидкість в межах норми ".

Цикл FOR в PHP

Основним завданням комп'ютерів є обробка великої кількості інформації, яка в людини зайняла б дуже багато часу. Для обробки таких завдань комп'ютер використовує цикли. Першим циклом яким ми почнемо розділ буде цикл for. Нижче наведено його синтаксис:

for (exp1; exp2; exp3) statement

У вираз exp1 вставляють початкове значення для лічильника циклу - змінна, яка вважає кількість разів виконання тіла циклу.

exp2 — задає умову повторення циклу. Цикл буде виконуватися доки ця умова буде true.

exp3 — виконується щоразу після виконання тіла циклу. Зазвичай воно використовується для зміни (збільшення або зменшення) лічильника.

Приклад:

```
<?php
   for ($i = 0; $i < 10; $i++)
    { echo "Вывод строки. 10 раз <br>"; }
?>
```

Відображення в браузері:

Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз

Цикли WHILE в PHP

Цикл WHILE замість використання лічильника циклу перевіряє деяку умову до того, поки це умова Істина (TRUE).

Синтаксис:

while (exp) statement

Умова перевіряється перед виконанням циклу, якщо вона буде хибною на початку, то цикл не виконається жодного разу! У тілі циклу повинна бути змінна яка впливатиме на умову, щоб запобігти зациклюванню. Приклад:

```
<?php
   $counter = 0;
   while ($counter < 5)
   { echo "Эта строка выведется 5 раз <br>"; $counter++; }
?>
```

Цикл DO... WHILE в PHP

Головна відмінність циклу DO ... WHILE від WHILE у тому, що перший спочатку виконується тіло циклу, а потім перевіряє умову. Тобто, якщо умова відразу Брехня, то цикл виконається один раз.

Синтаксис

do statement **while** (condition)

Використання циклу DO... WHILE:

```
<?php
   $counter = 6;
   do
      { echo "Эта строка выведется 1 раз <br>";
        $counter++;
       }
while ($counter < 5);
?>
```

Цей рядок виведеться 1 раз

Оскільки умова циклу відразу Брехня (6 > 5), цикл виконався лише один раз, оскільки спочатку виконується тіло циклу, та був перевіряється умова циклу.

Цикл FOREACH в PHP

Цикл FOREACH представлений спрощення роботи з масивами. Масиви складаються з окремих елементів, цикл FOREACH призначений для перебору цих елементів без лічильника.

Синтаксис:

foreach (**array as** $value) statement

foreach (**array as** $key => $value) statement

Використання циклу:

```
<?php
$array = array ("Apple", "Limon", "Chery", "Oranges");
foreach ($array as $value)
   { echo "Ви обрали фрукт - $value <br>"; }
?>
```

Відображення у браузері:

Ви вибрали фрукт - Apple
Ви вибрали фрукт - Limon
Ви вибрали фрукт - Chery
Ви вибрали фрукт - Oranges

Функції для обробки рядків у PHP

За допомогою цих функцій можна, наприклад, обрізати рядок, дописувати рядок, замінити частину рядка та багато іншого. Це дуже корисний інструмент для розробки скриптів.

Всі функції для обробки рядків наведені нижче:

Функція substr

Функція **substr** використовується для отримання частини рядка.

Синтаксис:

string substr (string $string, int $start [, int $length])

Перший параметр **$string** - рядок з якого потрібно отримати підрядок починаючи з позиції $start і довгою в $length.

Приклад:

```
<?php echo substr("Hello world", 6, 5); ?>
```

Відображення в браузері:

world

Останній параметр $length необовязковий

```
<?php echo substr("Hello world !!!", 6); ?>
```

Функція strpos

Функція повертає позицію першого входження підрядка в рядок

int strpos (string $string , mixed $needle [, int $offset = 0])

$string - рядок у якому буде зроблено пошук,

$needle - рядок, який потрібно знайти,

$offset - необов'язковий параметр, якщо цей параметр вказано, то пошук буде розпочато із зазначеної кількості символів з початку рядка

Приклад:

```
<?php
echo strpos("Hello world", "world"); // отримаємо 6
?>
```

В результаті отримаємо 6, тому що рядок "world" вперше зустрічається на 6 позиції

Створення масивів у PHP

Масив – це набір даних, які об'єднані під одним ім'ям.
Масив складається з кількох елементів, які мають певний індекс.
Масиви створюються за допомогою оператора присвоєння, так само, як і змінна.
Імена масивів починаються зі знака $, після якого слідує довільний ідентифікатор, далі йдуть квадратні дужки: **$**arr[0] = "php";
Дана конструкція створює масив і привласнює його елементу з індексом 0 значення "php", після чого ми можемо звертатися до цього елемента як до звичайної змінної: $ $ $ rr [0]. В результаті ми побачимо слово php.

Також ми можемо додати ще елементи до масиву:

```
<?php $arr[1] = "html"; $arr[2] = "css"; ?>
```

Як індекс елементів масиву ми можемо використовувати не тільки числа:

```
<?php
$arr["Kiev"] = 3000000;
$arr["Paris"] = 5000000;
$arr["LA"] = 15000000;
?>
```

Як значення індексів елементів і самих елементів ми можемо використовувати однакові типи даних **одночасно**!

Також існує скорочений запис для індексування:

```
<?php
$arr[] = 3000000; $arr[] = 5000000; $arr[] = 15000000;
?>
```

І тут перший елемент (3000000) отримає індекс 0! Потрібно мати це на увазі.

Для створення масиву можна використовувати функцію **array:**

```
<?php
$arr = array("php", "html", "css");
?>
```

У цьому випадку перший елемент отримає індекс 0. Якщо потрібно привласнити якийсь інший номер, можна скористатися конструкцією =>:

```
<?php
$arr = array(1 => "php", "html", "css"); ?>
```

Тепер елемент під номером 1 це "php", а чи не "html"! Також можна створити масив з рядковим індексом:

```
<?php
$arr = array("first" => "php", "second" => "html", "third" => "css");
?>
```

Починаючи з версії PHP 5.4, масиви можна створити через квадратні дужки:

```
<?php $arr = ["php", "laravel", "yii", "zend", "cakephp"]; ?>
```

Модифікація елементів масиву в PHP

Є масив:

```
<?php
$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";
?>
```

Для того, щоб змінити значення елемента використовуємо оператор присвоєння:

```
<?php
$arr[1] = "JAVASCRIPT";
?>
```

Для того, щоб додати новий елемент до кінця масиву використовує конструкцію:

```
<?php
$arr[] = "JQUERY";
?>
```

Для того, щоб вивести на екран масив можна використовувати **foreach:**

```
<?php
$arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS"; $arr[1] = "JAVASCRIPT"; $arr[] = "JQUERY";
foreach($arr as $key => $value) { // при переборе: $key - индекс элемента массива, $value - значение элемента массива
echo $value.'<br/>'; } ?>
```

Відображення у браузері:

PHP
JAVASCRIPT
CSS
JQUERY

Видалення елементів масиву в PHP

Якщо нам потрібно видалити один із елементів масиву, то для цього ми повинні використовувати функцію **unset**

```
<?php
$arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS";
unset($arr[1]);
foreach($arr as $key => $value)
                { echo $value.'<br/>'; }
?>
```

Перебір елементів масиву в PHP

Крім використання циклу для виведення всіх елементів масиву на екран ми можемо використовувати функцію print_r, яка виведе всі елементи масиву разом з їх індексами.

```
<?php
$arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS";
print_r($arr);
?>
```

Відображення у браузері:

```
Array ( [0] => PHP [1] => HTML [2] => CSS )
```

Також в PHP присутній спеціальний цикл для обробки масивів - цикл **foreach**

```php
<?php
$arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS";
foreach($arr as $value)
{
echo $value, "<br>";
}
?>
```

Відображення у браузері:

PHP
HTML
CSS

Для виведення індексу елемента потрібно використовувати другий варіант синтаксису циклу **foreach**

```
<?php
$arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS";
foreach($arr as $key => $value)
{
// $key - индекс эл.массива, $value - значение эл.массива
echo "[{$key}] => {$value} <br/>";
}
?>
```

Відображення у браузері:

[0] => PHP
[1] => HTML
[2] => CSS

Сортування масивів у PHP

Дуже часто потрібно відсортувати масив за індексом його елементів, за алфавітом його елементів, за зростанням, спаданням і т. д. У PHP для цього існують функції.

Функція - sort, яка сортує масив за зростанням значень його елементів, при цьому змінюючи індекс після сортування :

```
<?php
   $arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS";
   sort($arr);
   print_r($arr);
?>
```

Відображення у браузері

```
Array ( [0] => CSS    [1] => HTML    [2] => PHP )
```

Функція - **rsort**, яка сортує масив зі спадання значень його елементів, при цьому змінюючи індекс після сортування:

```
<?php
   $arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS";
   rsort($arr);
   print_r($arr);
?>
```

Відображення у браузері

```
Array ( [0] => PHP   [1] => HTML    [2] => CSS )
```

Третя функція -**ksort**, яка сортує масив за ключами, зберігаючи відносини між ключами та значеннями:

```
<?php
  $arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS"; ksort($arr);
  print_r($arr); ?>
```

Відображення у браузері:

```
Array ( [0] => PHP   [1] => HTML  [2] => CSS )
```

Функція - **krsort**, яка сортує масив зі спадання індексів його елементів:

```
<?php
   $arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS";
   krsort($arr);
   print_r($arr);
?>
```

Відображення у браузері:

Array ([2] => CSS [1] => HTML [0] => PHP)

Навігація по масивам у PHP

Навігація по масиву дає можливість дізнатися поточний, наступний, попередній, останній елемент масиву.

Поточний елемент масиву визначає функцію current:

```
<?php echo "Now is: ", current($arr), "<br>"; ?>
```

Наступний елемент масиву визначає функцію next:

```
<?php echo "Next is: ", next($arr), "<br>"; ?>
```

Попередній елемент масиву визначає функцію prev:

```
<?php echo "Previously is: ", prev($arr), "<br>"; ?>
```

Останній елемент масиву визначає функцію end:

```
<?php echo "The end is: ", end($arr), "<br>"; ?>
```

Для визначення першого (**повернення покажчика**) елемента масиву використовують функцію reset:

```
<?php echo "First is: ", reset($arr), "<br>"; ?>
```

Приклад навігації масивами:

```
<?php $arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS";
   echo «Даний_елемент: ", current($arr), "<br>";
   echo «Наст_елемент:  ", next($arr), "<br>";
   echo "Попередній_елемент: ", prev($arr), "<br>";
   echo «Останній_елемент: ", end($arr), "<br>";
   echo "Перший_елемент: ", reset($arr), "<br>";
?>
```

Відображення у браузері:

Даний_елемент: PHP
Наст_елемент: HTML
Попередній_елемент: PHP
Останній_елемент: CSS
Перший_елемент: PHP

Перетворення рядків на масиви і навпаки

PHP вміє перетворювати дані з рядка в масив і навпаки, для цього в PHP є функція implode і explode.
implode - формує рядок з масиву.
explode - формує масив з рядка.
Використання функції **implode:**

```
<?php
$arr[0] = "PHP"; $arr[1] = "HTML"; $arr[2] = "CSS";
$string = implode(", ", $arr);
echo $string;
?>
```

Відображення у браузері:

PHP, HTML, CSS

Використання функції **explode:**

```
<?php
  $string = "PHP, HTML, CSS";
  $arr = explode(", ", $string);
  print_r($arr);
?>
```

Відображення у браузері:

Array ([0] => PHP [1] => HTML [2] => CSS)