Работа с математическими функциями в PHP

Изучите методы и функции: abs, sqrt, pow, round, ceil, floor, min, max, mt_rand.

Функция abs

Функция **abs** вычисляет модуль числа (то есть из отрицательного делает положительное).

Синтаксис

```
abs(число);
```

Примеры

Пример

В данном примере функция вычислила модуль от числа **-15**:

```
<?php
        echo abs(-15);
?>
```

Результат выполнения кода:

```
15
```

Функция sqrt

Функция **sqrt** находит квадратный корень числа.

См. также функцию pow, которая возводит число в степень.

См. также функцию round, которая округляет число по правилам математического округления.

Синтаксис

```
sqrt(число);
```

Примеры

Пример

В данном примере функция вычислит квадратный корень 16:

```
<?php
        echo sqrt(16);
?>
```

Результат выполнения кода:

```
4
```

Пример

В данном примере функция вычислит квадратный корень 5:

```
<?php
        echo sqrt(5);
?>
```

Результат выполнения кода:

```
2.2360679774998
```

Функция pow

Функция **pow** возводит число в заданную степень.

См. также функцию sqrt, которая вычисляет квадратный корень числа.

Синтаксис

```
pow(число, степень);
```

Примеры

Пример

В данном примере функция возведет число 2 в степень 3:

```
<?php
        echo pow(2, 3);
?>
```

Результат выполнения кода:

```
8
```

Функция round

Функция **round** округляет число по правилам математического округления.

См. также функцию ceil, которая округляет дробь в большую сторону.

См. также функцию floor, которая округляет дробь в меньшую сторону.

Синтаксис

```
round(число, [сколько знаков оставить в дробной части]);
```

Второй необязательный параметр может быть как положительным, так и отрицательным (в этом случае он указывает сколько знаков оставить в целой части).

Примеры

Пример

Округлим дробь до целого:

```
<?php
        echo round(3.4);
?>
```

Результат выполнения кода:

```
3
```

Пример

Округлим дробь до целого:

```
<?php
        echo round(3.5);
?>
```

Результат выполнения кода:

```
4
```

Пример

Округлим дробь до целого:

```
<?php
        echo round(3.6);
?>
```

Результат выполнения кода:

```
4
```

Пример

В данном примере функция округлила число до двух знаков в дробной части:

```
<?php
        echo round(12.45678, 2);
?>
```

Результат выполнения кода:

```
12.46
```

Пример

В данном примере функция округлила число до трех знаков в дробной части:

```
<?php
        echo round(12.45678, 3);
?>
```

Результат выполнения кода:

```
12.457
```

Функция ceil

Функция **ceil** округляет дробь в большую сторону до целого.

Это значит, что независимо от цифры в начале дробной части, дробь все равно округлится с увеличением целой части на 1.

К примеру, **12.1** округлится к **13**.

См. также функцию round, которая округляет дробь по правилам математического округления.

См. также функцию floor, которая округляет дробь в меньшую сторону.

Синтаксис

```
ceil(число);
```

Примеры

Пример

В данном примере функция округлила 5.1 в большую сторону и вернула 6:

```
<?php
        echo ceil(5.1);
?>
```

Результат выполнения кода:

```
6
```

Функция floor

Функция **floor** округляет дробь в меньшую сторону.

Это значит, что независимо от цифры в начале дробной части, дробь все равно округлится без увеличения целой части на 1.

К примеру, **12.9** округлится к **12**.

См. также функцию round, которая округляет дробь по правилам математического округления.

См. также функцию ceil, которая округляет дробь в большую сторону.

Синтаксис

```
floor(число);
```

Примеры

Пример

В данном примере функция округлила 4.9 в меньшую сторону и вернула 4:

```
<?php
        echo floor(4.9);
?>
```

Результат выполнения кода:

```
4
```

Функция min

Функция **min** находит самое маленькое число из переданных ей параметрами или самое маленькое число среди элементов массива.

См. также функцию max, которая находит наибольшее число.

Синтаксис

```
min(первое число, второе число....);

min(массив чисел);
```

Примеры

Пример

В данном примере функция вернет наименьшее среди заданных чисел:

```
<?php
echo min(1, 2, 3);?>
```

Результат выполнения кода:

```
1
```

Пример

В данном примере функция вернет наименьшее значение массива:

```
<?php
echo min([1, 2, 3]);?>
```

Результат выполнения кода:

```
1
```

Функция max

Функция **max** находит самое большое число из переданных ей параметрами или самое большое число среди элементов массива.

См. также функцию min, которая находит наименьшее число.

Синтаксис

```
max(первое число, второе число....);

max(массив чисел);
```

Примеры

Пример

В данном примере функция вернет наибольшее среди заданных чисел:

```
<?php
echo max(1, 2, 3);?>
```

Результат выполнения кода:

```
3
```

Пример

В данном примере функция вернет наибольшее значение массива:

```
<?php
echo max([1, 2, 3]);?>
```

Результат выполнения кода:

```
3
```

Функция mt_rand

Функция **mt_rand** генерирует случайное целое число в заданном промежутке.

Вам также может пригодиться функция **mt_getrandmax**, которая возвращает наибольшее возможное случайное значение числа.

Синтаксис

```
mt_rand(с какого числа, до какого числа);
```

Примеры

Пример

В данном примере функция сгерерирует случайное число от 5 до 15 (при обновлении страницы число каждый раз будет разным):

```
<?php
        echo mt_rand(5, 15);
?>
```

Результат выполнения кода:

```
13
```

Задачи на математические функции PHP

Примеры решения задач

Задача . Округление и ассоциативный массив

Задача. Найдите корень из числа **1000**. Округлите его в большую и меньшую стороны. В массив **$arr**запишите первым элементом корень из числа, вторым элементом - округление в меньшую сторону, третьим элементом - в большую.

Решение: корень из числа найдем функцией sqrt. Далее, чтобы округлить число в **меньшую** строну, воспользуемся функцией floor, а чтобы в большую - функцией ceil:

```
<?php
        $sqrt = sqrt(1000); //найдем корень и запишем его в $sqrt

        echo floor($sqrt); //округлим в меньшую сторону
        echo ceil($sqrt); //округлим в большую сторону
?>
```

Теперь результаты необходимо записать в массив. Сделать это можно **двумя** способами: объявить через **[ ]** либо просто воспользоваться присваиванием **$arr[ ] = 'html'; $arr[ ] = 'php';** и так далее.

Первый способ:

```
<?php
        $sqrt = sqrt(1000)
        $arr = [$sqrt, floor($sqrt], ceil($sqrt));
?>
```

Второй способ:

```
<?php
        $sqrt = sqrt(1000)
        $arr[] = $sqrt;
        $arr[] = floor($sqrt);
        $arr[] = ceil($sqrt);
?>
```

Задача . Массив случайных чисел

Задача. Заполните массив 30-ю случайными числами от 1 до 10.

Решение: для решения воспользуемся циклом **for** - прокрутим его **30** раз, записывая при каждом проходе случайное число в **новый элемент массива**.

Случайные числа будем получать через функцию mt_rand.

Чтобы число записалось в новый элемент массива, следует сделать так: **$arr[ ] = 1**; **$arr[ ] = 2**; - первое число запишется в нулевой элемент массива, а второе - в первый (с ключом **1**). В случае со случайными числами это будет выглядеть так:

```php
<?php
	//Каждое из чисел будет записываться в новый элемент массива:
	$arr[] = mt_rand(1, 10);
	$arr[] = mt_rand(1, 10);
	$arr[] = mt_rand(1, 10);
?>
```

Напоминаю о том, что **ключи можно не оставлять пустыми**, а делать своими:

```php
<?php
	//Используем свои ключи, а не автоматические:
	$arr['Первый ключ'] = mt_rand(1, 10);
	$arr['Второй ключ'] = mt_rand(1, 10);
	$arr['Третий ключ'] = mt_rand(1, 10);
?>
```

При большом желании мы можем **вручную** сделать 30 таких строчек - и задача решена. Но пусть **лучше** за нас это сделает цикл:

```php
<?php
	//Переменная $i нужна, чтобы цикл сделал 30 итераций (проходов)
	for ($i = 1; $i <= 30; $i++) {
		$arr[] = mt_rand(1, 10);
	}

	var_dump($arr);

	/*
		Пообновляйте страницу -
		вы увидите как меняется массив,
		так как он заполняется случайными числами.
	*/
?>
```

Задачи для решения

Работа с %

1. Даны переменные **$a=10** и **$b=3**. Найдите остаток от деления **$a** на **$b**. Показать решение.

2. Даны переменные **$a** и **$b**. Проверьте, что **$a** делится без остатка на **$b**. Если это так - выведите 'Делится' и результат деления, иначе выведите 'Делится с остатком' и остаток от деления. Показать решение.

Работа со степенью и корнем

Для решения задач данного блока вам понадобятся следующие функции: sqrt, pow.

3. Возведите **2** в **10** степень. Результат запишите в переменную **$st**.

4. Найдите квадратный корень из **245**.

5. Дан массив с элементами **4, 2, 5, 19, 13, 0, 10**. Найдите корень из суммы квадратов его элементов. Для решения воспользуйтесь циклом **foreach**.

Работа с функциями округления

Для решения задач данного блока вам понадобятся следующие функции: round, ceil, floor.

6. Найдите квадратный корень из **379**. Результат округлите до целых, до десятых, до сотых.

7. Найдите квадратный корень из **587**. Округлите результат в большую и меньшую сторону, запишите результаты округления в ассоциативный массив с ключами **'floor'** и **'ceil'**.

Работа с min и max

Для решения задач данного блока вам понадобятся следующие функции: min, max.

8. Даны числа **4, -2, 5, 19, -130, 0, 10**. Найдите минимальное и максимальное число.

Работа с рандомом

Для решения задач данного блока вам понадобятся следующие функции: mt_rand.

9. Выведите на экран случайное число от **1** до **100**.

10. Заполните массив 10-ю случайными числами. *Подсказка: нужно воспользоваться циклами for или while.* Показать решение.

Работа с модулем

Для решения задач данного блока вам понадобятся следующие функции: abs.

11. Даны переменные **$a** и **$b**. Найдите найдите модуль разности **$a** и **$b**. Проверьте работу скрипта самостоятельно для различных **$a** и **$b**.

12. Дан массив в числами, к примеру **[1, 2, -1, -2, 3, -3]**. Создайте из него новый массив так, чтобы отрицательные числа стали положительными, то есть у нас должен получиться такой массив: **[1, 2, 1, 2, 3, 3]**.

Задачи

13. Дано число, например **30**. У этого числа есть делители - числа, на которое оно делится без остатка. Делители числа 30 - это 1, 2, 3, 5, 6, 10, 15, 30. Задача: сделайте массив делителей нашего числа. Число может быть любым, не обязательно, 30.

14. Дан массив [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. Узнайте, сколько первых элементов массива нужно сложить, чтобы сумма получилась больше **10**.