PHP Math Operations

In this tutorial you will learn how to perform mathematical operations in PHP.

Performing Math Operations

PHP has several built-in functions that help you perform anything from simple additions or subtraction to advanced calculations. You've already seen how to perform basic mathematical operations in PHP operators chapter. Let's check out one more example:

Example **Run this code »**

```php
<?php
echo 7 + 3; // 0utputs: 10
echo 7 - 2; // 0utputs: 5
echo 7 * 2; // 0utputs: 14
echo 7 / 2; // 0utputs: 3.5
echo 7 % 2; // 0utputs: 1
?>
```

Every math operation has a certain precedence level; generally multiplication and division are performed before addition and subtraction. However, parentheses can alter this precedence; expressions enclosed within parentheses are always evaluated first, regardless of the operation's precedence level, as demonstrated in the following example:

Example **Run this code »**

```php
<?php
echo 5 + 4 * 10;          // 0utputs: 45
echo (5 + 4) * 10;        // 0utputs: 90
echo 5 + 4 * 10 / 2;      // 0utputs: 25
echo 8 * 10 / 4 - 2;      // 0utputs: 18
echo 8 * 10 / (4 - 2);    // 0utputs: 40
echo 8 + 10 / 4 - 2;      // 0utputs: 8.5
echo (8 + 10) / (4 - 2); // 0utputs: 9
?>
```

In the following section we're going to look at some built-in PHP functions that are most frequently used in performing mathematical operations.

Find the Absolute Value of a Number

The absolute value of an integer or a float can be found with the `abs()` function, as demonstrated in the following example:

Example **Run this code »**

```php
<?php
echo abs(5);    // Outputs: 5 (integer)
echo abs(-5);   // Outputs: 5 (integer)
echo abs(4.2);  // Outputs: 4.2 (double/float)
echo abs(-4.2); // Outputs: 4.2 (double/float)
?>
```

As you can see if the given number is negative, the valued returned is positive. But, if the number is positive, this function simply returns the number.

Round a Fractional Value Up or Down

The `ceil()` function can be used to round a fractional value up to the next highest integer value, whereas the `floor()` function can be used to round a fractional value down to the next lowest integer value, as demonstrated in the following example:

Example **Run this code »**

```php
<?php
// Round fractions up
echo ceil(4.2);    // Outputs: 5
echo ceil(9.99);   // Outputs: 10
echo ceil(-5.18);  // Outputs: -5

// Round fractions down
echo floor(4.2);    // Outputs: 4
echo floor(9.99);   // Outputs: 9
echo floor(-5.18);  // Outputs: -6
?>
```

Find the Square Root of a Number

You can use the `sqrt()` function to find the square root of a positive number. This function returns a special value `NAN` for negative numbers. Here's an example:

Example **Run this code »**

```php
<?php
echo sqrt(9);   // Outputs: 3
echo sqrt(25);  // Outputs: 5
echo sqrt(10);  // Outputs: 3.1622776601684
echo sqrt(-16); // Outputs: NAN
?>
```

Generate a Random Number

The `rand()` function can be used to generate a random number. You can optionally specify a range by passing the min, max arguments, as shown in the following example:

Example **Run this code »**

```php
<?php
// Generate some random numbers
echo rand() . "<br>";
echo rand() . "<br>";

// Generate some random numbers between 1 and 10 (inclusive)
echo rand(1, 10) . "<br>";
echo rand(1, 10) . "<br>";
?>
```

If `rand()` function is called without the optional **`min`**, **`max`** arguments, it returns a pseudo-random number between `0` and `getrandmax()`. The `getrandmax()` function show the largest possible random value, which is only 32767 on Windows platform. So, if you require a range larger than 32767, you may simply specify the **`min`** and **`max`** arguments.

Convert Decimal Numbers to Binary and Vice Versa

The `decbin()` function is used to convert a decimal number into binary number. Whereas its counterpart the `bindec()` function converts a number from binary to decimal.

Example **Run this code »**

```php
<?php
// Convert Decimal to Binary
echo decbin(2);    // Outputs: 10
echo decbin(12);   // Outputs: 1100
echo decbin(100);  // Outputs: 1100100

// Convert Binary to Decimal
echo bindec(10);       // Outputs: 2
echo bindec(1100);     // Outputs: 12
echo bindec(1100100);  // Outputs: 100
?>
```

Convert Decimal Numbers to Hexadecimal and Vice Versa

The `dechex()` function is used to convert a decimal number into hexadecimal representation. Whereas, the `hexdec()` function is used to converts a hexadecimal string to a decimal number.

Example **Run this code »**

```php
<?php
// Convert decimal to hexadecimal
echo dechex(255);  // Outputs: ff
echo dechex(196);  // Outputs: c4
echo dechex(0);    // Outputs: 0

// Convert hexadecimal to decimal
echo hexdec('ff');  // Outputs: 255
echo hexdec('c4');  // Outputs: 196
echo hexdec(0);     // Outputs: 0
?>
```

Convert Decimal Numbers to Octal and Vice Versa

The `decoct()` function is used to convert a decimal number into octal representation. Whereas, the `octdec()` function is used to converts a octal number to a decimal number.

Example **Run this code »**

```php
<?php
// Convert decimal to octal
echo decoct(12);   // Outputs: 14
echo decoct(256);  // Outputs: 400
echo decoct(77);   // Outputs: 115

// Convert octal to decimal
echo octdec('14');   // Outputs: 12
echo octdec('400');  // Outputs: 256
echo octdec('115');  // Outputs: 77
?>
```

Convert a Number from One Base System to Another

The `base_convert()` function can be used to convert a number from one base system to other. For example, you can convert decimal (*base 10*) to binary (*base 2*), hexadecimal (*base 16*) to octal (*base 8*), octal to hexadecimal, hexadecimal to decimal, and so on.

This function accepts three parameters: the number to convert, the base it's currently in, and the base it's to be converted to. The basic syntax is as follows:

```php
base_convert(number, frombase, tobase);
```

Here, the number can be either an integer or a string representing an integer.
Both **`frombase`** and **`tobase`** have to be between 2 and 36, inclusive. Digits in numbers with a base higher than 10 will be represented with the letters a-z, where a means 10, b means 11 and z means 35. Here's a simple example to show how this function works:

Example **Run this code »**

```php
<?php
// Convert decimal to binary
echo base_convert('12', 10, 2);  // Outputs: 1100
// Convert binary to decimal
echo base_convert('1100', 2, 10);  // Outputs: 12

// Convert decimal to hexadecimal
echo base_convert('10889592', 10, 16);  // Outputs: a62978
// Convert hexadecimal to decimal
echo base_convert('a62978', 16, 10);  // Outputs: 10889592

// Convert decimal to octal
```

```php
echo base_convert('82', 10, 8);  // Outputs: 122
// Convert octal to decimal
```

Note: The `base_convert()` function will always return a string value. If the returned value is in base 10 the resulting string can be used as a numeric string in calculations and PHP will convert it to a number when the calculation is performed.