

PHP

Math: Functions and constants, MD5

Primary source: <https://www.w3schools.com/php/>

Internet Programming 2, Lesson 6

April 2024

PHP Math
Random
MD5

PHP Math Introduction

The math functions can handle values within the range of integer and float types.

Installation

The PHP math functions are part of the PHP core. No installation is required to use these functions.

PHP Math Functions

The **floor()** function rounds a number DOWN to the nearest integer, if necessary. **floor(number);**

The **ceil()** function rounds a number UP to the nearest integer, if necessary. **ceil(number);**

The **round()** function rounds a floating-point number. **round(number[, precision], [const]);** precision = number of decimal places, default is 0.

Examples – ouptup indicated in green text below:

```
<?php
echo(floor(0.60) . "<br>");    0
echo(floor(0.40) . "<br>");    0
echo(floor(5) . "<br>");        5
echo(floor(5.1) . "<br>");      5
echo(floor(-5.1) . "<br>");     -6
echo(floor(-5.9));            -6
?>
```

```
<?php
echo(ceil(0.60) . "<br>");      1
echo(ceil(0.40) . "<br>");      1
echo(ceil(5) . "<br>");          5
echo(ceil(5.1) . "<br>");        6
echo(ceil(-5.1) . "<br>");       -5
echo(ceil(-5.9));             -5
?>
```

```
<?php
echo(round(0.60) . "<br>");      1
echo(round(0.50) . "<br>");      1
echo(round(0.49) . "<br>");      0
echo(round(-4.40) . "<br>");     -4
echo(round(-4.60));            -5
?>
```

The **base_convert()** function converts a number from one number base to another. **base_convert(*number,frombase,tobase*);**

```
<?php
$hex = "E196";
echo base_convert($hex,16,8);           160626
?>
```

The **fmod()** function returns the remainder (modulo) of x/y. **fmod(*x,y*);**

```
<?php
$x = 7;
$y = 2;
$result = fmod($x,$y);
echo $result;                           1
// $result equals 1, because 2 * 3 + 1 = 7
?>
```

The **is_finite()** function checks whether a value is finite or not.

This function returns true (1) if the specified value is a finite number, otherwise it returns false/nothing.

The value is finite if it is within the allowed range for a PHP float on this platform.

is_finite(value);

```
<?php
echo is_finite(2) . "<br>";           1
echo is_finite(log(0)) . "<br>";
echo is_finite(2000);                1
?>
```

The **is_infinite()** function checks whether a value is infinite or not.

This function returns true (1) if the specified value is an infinite number, otherwise it returns false/nothing.

The value is infinite if it is outside the allowed range for a PHP float on this platform.

is_infinite(value);

```
<?php
echo is_infinite(2) . "<br>";
echo is_infinite(log(0)) . "<br>";    1
echo is_infinite(2000);
?>
```

The **max()** function returns the highest value in an array, or the highest value of several specified values.

max(array_values);

or

max(value1,value2,...);

Parameter	Description
<i>array_values</i>	Required. Specifies an array containing the values
<i>value1,value2,...</i>	Required. Specifies the values to compare (must be at least two values)

```
<?php
echo(max(2,4,6,8,10) . "<br>");      10
echo(max(22,14,68,18,15) . "<br>");  68
echo(max(array(4,6,8,10)) . "<br>");  10
echo(max(array(44,16,81,12)));      81
?>
```

The **min()** function returns the lowest value in an array, or the lowest value of several specified values.

min(array_values);

or

min(value1,value2,...);

Parameter	Description
-----------	-------------

<i>array_values</i>	Required. Specifies an array containing the values
---------------------	--

<i>value1,value2,...</i>	Required. Specifies the values to compare (must be at least two values)
--------------------------	---

```
<?php
echo(min(2,4,6,8,10) . "<br>");      2
echo(min(22,14,68,18,15) . "<br>");  14
echo(min(array(4,6,8,10)) . "<br>");  4
echo(min(array(44,16,81,12)));      12
?>
```

The **is_nan()** function checks whether a value is 'not a number'.

This function returns true (1) if the specified value is 'not a number', otherwise it returns false/nothing.

```
is_nan(value);      <?php
                    echo is_nan(200) . "<br>";
                    echo is_nan(acos(1.01));      1
                    ?>
```

The **lcg_value()** function returns a pseudo random number in a range between 0 and 1.

```
lcg_value();

                    <?php
                    echo lcg_value();
                    echo "<br>";
                    echo lcg_value();
                    ?>      0.972903666993
                        0.4035136134451
```

lcg_value()



Run >

Result Size: 355 x 462

Get your website

```
<!DOCTYPE html>
<html>
<body>

<?php
for ($i=0; $i <=10; $i++){
    echo lcg_value();
    echo "<br>";
    echo lcg_value(); }
?>

</body>
</html>
```

```
0.50587455956741
0.405664129529440.87268601734223
0.374687971515350.82077117499056
0.524930403440080.68775414594268
0.118625605452560.23497930716443
0.619801106992480.49011613460622
0.138818846142590.31706257050789
0.00649590342315980.24918787301781
0.676456875444790.85522884645447
0.924754090201280.30896906279251
0.246142102595120.58361858369819
0.93506489914464
```


rand(): Definition and Usage

The rand() function generates a random integer.

Example tip: If you want a random integer between 10 and 100 (inclusive), use rand (10,100).

Tip: As of PHP 7.1, the rand() function has been an alias of the [mt_rand\(\)](#) function.

Syntax

rand(); or rand(*min*,*max*);

```
<?php
echo(rand() . "<br>");
echo(rand() . "<br>");
echo(rand(10,100));
?>
```

14730
22647
95

Parameter Values

Parameter	Description
<i>min</i>	Optional. Specifies the lowest number to return. Default is 0
<i>max</i>	Optional. Specifies the highest number to return. Default is getrandmax()

Technical Details

Return Value:	A random integer between <i>min</i> (or 0) and <i>max</i> (or getrandmax()) inclusive)
Return Type:	Integer
PHP Version:	4+
PHP Changelog:	PHP 7.1: The rand() function is an alias of mt_rand() . PHP 4.2.0: The random number generator is seeded automatically.

mt_rand(): Definition and Usage

The mt_rand() function generates a random integer using the Mersenne Twister algorithm.

Example tip: If you want a random integer between 10 and 100 (inclusive), use mt_rand(10,100).

Syntax

```
mt_rand();           or           mt_rand(min,max);
```

```
<?php
echo(mt_rand() . "<br>");           793546867
echo(mt_rand() . "<br>");           446702575
echo(mt_rand(10,100));             91
?>
```

Parameter Values

Parameter	Description
<i>min</i>	Optional. Specifies the lowest number to return. Default is 0
<i>max</i>	Optional. Specifies the highest number to return. Default is mt_getrandmax()

Technical Details

Return Value:	A random integer between <i>min</i> (or 0) and <i>max</i> (or mt_getrandmax() inclusive). Returns FALSE if <i>max</i> < <i>min</i>
Return Type:	Integer
PHP Version:	4+
PHP Changelog:	PHP 7.1: rand() has been an alias of mt_rand() PHP 5.3.4: Issues an E_WARNING and returns FALSE if <i>max</i> < <i>min</i> . PHP 4.2.0: Random number generator is seeded automatically.

Important Math Functions

Function	Description
<u>abs()</u>	Returns the absolute (positive) value of a number
<u>base_convert()</u>	Converts a number from one number base to another
<u>bindec()</u>	Converts a binary number to a decimal number
<u>ceil()</u>	Rounds a number up to the nearest integer
<u>decbin()</u>	Converts a decimal number to a binary number
<u>dechex()</u>	Converts a decimal number to a hexadecimal number
<u>decoct()</u>	Converts a decimal number to an octal number
<u>deg2rad()</u>	Converts a degree value to a radian value
<u>floor()</u>	Rounds a number down to the nearest integer
<u>fmod()</u>	Returns the remainder of x/y
<u>getrandmax()</u>	Returns the largest possible value returned by rand()
<u>hexdec()</u>	Converts a hexadecimal number to a decimal number
<u>hypot()</u>	Calculates the hypotenuse of a right-angle triangle
<u>intdiv()</u>	Performs integer division
<u>is_finite()</u>	Checks whether a value is finite or not
<u>is_infinite()</u>	Checks whether a value is infinite or not
<u>is_nan()</u>	Checks whether a value is 'not-a-number'
<u>lcg_value()</u>	Returns a pseudo random number in a range between 0 and 1

Important Math Functions...Cont.

Function	Description
<u>max()</u>	Returns the highest value in an array, or the highest value of several specified values
<u>min()</u>	Returns the lowest value in an array, or the lowest value of several specified values
<u>mt_getrandmax()</u>	Returns the largest possible value returned by mt_rand()
<u>mt_rand()</u>	Generates a random integer using Mersenne Twister algorithm
<u>mt_srand()</u>	Seeds the Mersenne Twister random number generator
<u>octdec()</u>	Converts an octal number to a decimal number
<u>pi()</u>	Returns the value of PI
<u>pow()</u>	Returns x raised to the power of y
<u>rad2deg()</u>	Converts a radian value to a degree value
<u>rand()</u>	Generates a random integer
<u>round()</u>	Rounds a floating-point number
<u>sqrt()</u>	Returns the square root of a number
<u>srand()</u>	Seeds the random number generator

The **pi()** function returns the value of PI.

Tip: The named constant **M_PI** is identical to **pi()**.

```
pi();  
  
<?php  
echo(pi());  
?>
```

3.1415926535898

The **pow()** function returns x raised to the power of y.

```
pow(x,y);  
  
<?php  
echo(pow(2,4) . "<br>");  
echo(pow(-2,4) . "<br>");  
echo(pow(-2,-4) . "<br>");  
echo(pow(-2,-3.2));  
?>
```

16
16
0.0625
NAN

Parameter	Description
x	Required. Specifies the base to use
y	Required. Specifies the exponent

PHP Predefined Math Constants

M_PI	3.14159265358979323846	Returns Pi	PHP 4
M_PI_2	1.57079632679489661923	Returns Pi/2	PHP 4
M_PI_4	0.78539816339744830962	Returns Pi/4	PHP 4
M_1_PI	0.31830988618379067154	Returns 1/Pi	PHP 4
M_2_PI	0.63661977236758134308	Returns 2/Pi	PHP 4

INF	INF	The infinite
-----	-----	--------------

NAN	NAN	Not A Number
-----	-----	--------------

MD5 Encryption

Definition and Usage

The md5() function calculates the MD5 hash of a string.

The md5() function uses the RSA Data Security, Inc. MD5 Message-Digest Algorithm.

From RFC 1321 - The MD5 Message-Digest Algorithm: *"The MD5 message-digest algorithm takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input. The MD5 algorithm is intended for digital signature applications, where a large file must be "compressed" in a secure manner before being encrypted with a private (secret) key under a public-key cryptosystem such as RSA."*

To calculate the MD5 hash of a file, use the [md5_file\(\)](#) function.

Syntax: md5(string,raw)

Parameter	Description
<i>string</i>	Required. The string to be calculated
<i>raw</i>	<ul style="list-style-type: none">•Optional. Specifies hex or binary output format: TRUE - Raw 16 character binary format•FALSE - Default. 32 character hex number

Example:

```
<?php
$str = "Hello";
echo md5($str);
?>
```

Output:

8b1a9953c4611296a827abf8c47804d7

What does MD5 mean?

MD5 is the abbreviation of 'Message-Digest algorithm 5'.

The MD5 algorithm is used as an encryption or fingerprint function for a file.

Often used to encrypt database passwords, MD5 is also able to generate a file thumbprint to ensure that a file is identical after a transfer for example.

An MD5 hash is composed of 32 hexadecimal characters.

Enter a word in the MD5 encryption form above to know the corresponding MD5 hash

[source: <https://www.md5online.org/md5-encrypt.html>]

```
<?php
```

```
$str = "Hello";
```

```
echo md5($str);
```

```
if (md5($str) == "8b1a9953c4611296a827abf8c47804d7")
```

```
{
```

```
    echo "<br>Hello world!";
```

```
    exit;
```

```
}
```

```
?>
```

8b1a9953c4611296a827abf8c47804d7

Hello world!

PHP md5_file() Function

Example

Calculate the MD5 hash of the text file "test.txt":

```
<?php
$filename = "test.txt";
$md5file = md5_file($filename);
echo $md5file;
?>
```

The output of the code above will be:

d41d8cd98f00b204e9800998ecf8427e

Definition and Usage

The md5_file() function calculates the MD5 hash of a file.

The md5_file() function uses the RSA Data Security, Inc. MD5 Message-Digest Algorithm.

From RFC 1321 - The MD5 Message-Digest Algorithm: *"The MD5 message-digest algorithm takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input. The MD5 algorithm is intended for digital signature applications, where a large file must be "compressed" in a secure manner before being encrypted with a private (secret) key under a public-key cryptosystem such as RSA."*

To calculate the MD5 hash of a string, use the [md5\(\)](#) function.

Syntax

`md5_file(file,raw)`

Parameter Values

Parameter	Description
<i>file</i>	Required. The file to be calculated
<i>raw</i>	<ul style="list-style-type: none">•Optional. A boolean value that specifies hex or binary output format:TRUE - Raw 16 character binary format•FALSE - Default. 32 character hex number

Technical Details

Return Value:	Returns the calculated MD5 hash on success, or FALSE on failure
PHP Version:	4.2.0+
Changelog:	<p>The <i>raw</i> parameter was added in PHP 5.0</p> <p>As of PHP 5.1, it is possible to use md5_file() with wrappers, e.g. <code>md5_file("https://w3schools.com/..")</code></p>

Exercise

1. Write a PHP program to perform the following tasks: save the solution as **passwords.php**
 - 1.1. Create an associative array called \$passwords;
 - 1.2. populate the above array with 5 randomly-generated and md5-encrypted passwords – use the randomly-generated password as the key (index) and the encrypted form of the password as the content of the array elements – see below for rules pertaining to password formulation.
 - 1.3. Display the actual password and it's encrypted form in a neat table on the web page.
Generate the passwords as follows: [write a function to generate the password]
 - Minimum length is 3 and maximum length is 8 characters;
 - Passwords may contain lowercase letters and/or the digits 0 to 9 only;
 - Passwords cannot start with a number.
2. Write a PHP script to perform the following tasks: save the solution as **accessControl.php**
 - 2.1. Create an associative array called \$users;
 - 2.2. populate the array with a minimum of 5 elements as follows:

userID	Password
Ramdeyal	ram123
Dlamini	dla123
Zuma	zum123

...and so on. Remember to encrypt the password using the md5() function.
 - 2.3. provide a log-in form for the user, and control access as follows: if the userID does not exist, display the message “user unknown – please check your userid before trying again” and allow the user to try to login again. If the password is correct, Display “Access granted” and donot display the login for again. If the password is incorrect, allow 3 chances for the user to login. Once 3 incorrect passwords have been submitted, block the user and display “Access blocked: please consult with your administrator”.