PHP LAB 1

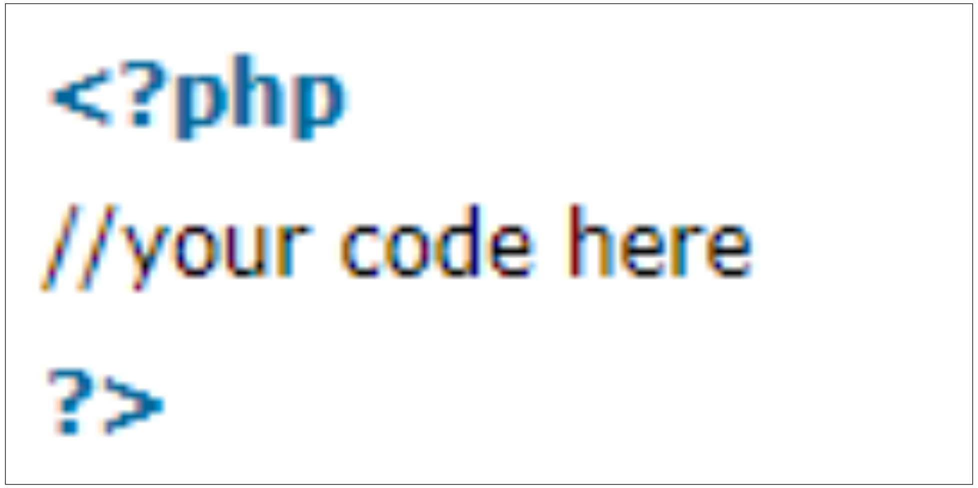
SAMITHA SOMATHILAKA

Department of Computing & Mathematics, WIT

PHP ECHO

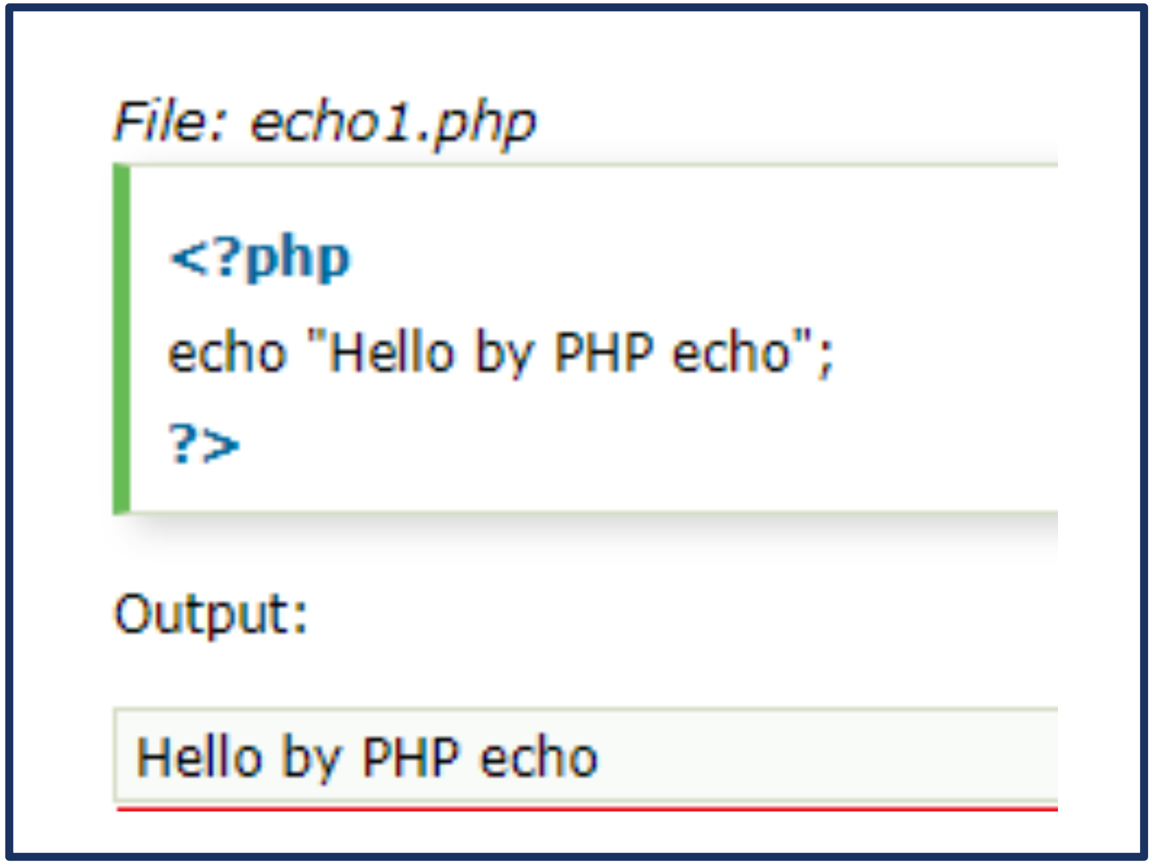
PHP EXAMPLE

- It is very easy to create a simple PHP example.
- To do so, create a file and write HTML tags + PHP code and save this file with .php extension.
- All PHP code goes between php tag. A syntax of PHP tag is given below:



```
<?php
//your code here
?>
```

PHP ECHO



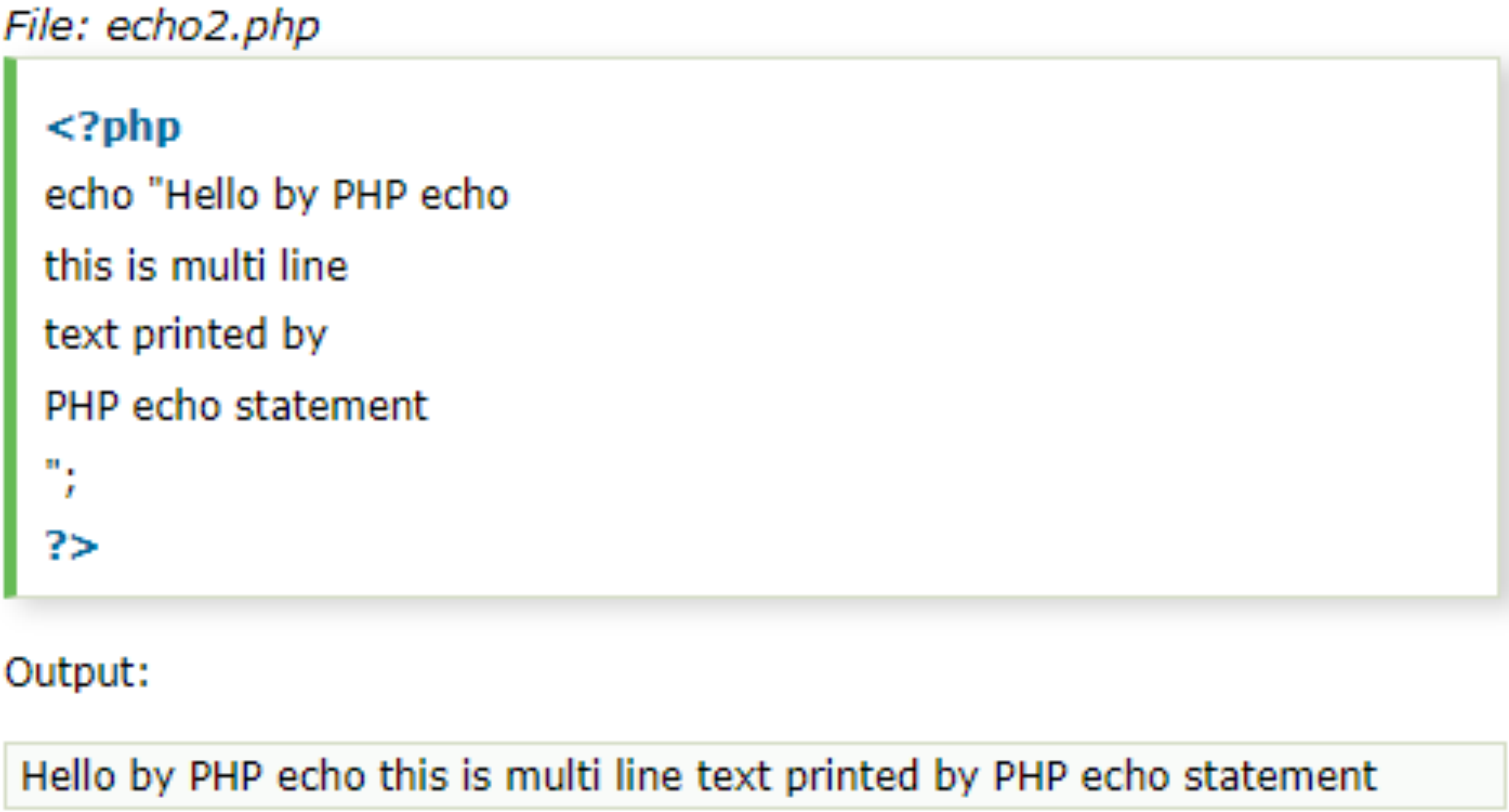
File: echo1.php

```
<?php
echo "Hello by PHP echo";
?>
```

Output:

```
Hello by PHP echo
```

- PHP echo is a language construct not a function, so you don't need to use parenthesis with it.
- But if you want to use more than one parameters, it is required to use parenthesis.
- PHP echo statement can be used to print string, multi line strings, escaping characters, variable, array etc.

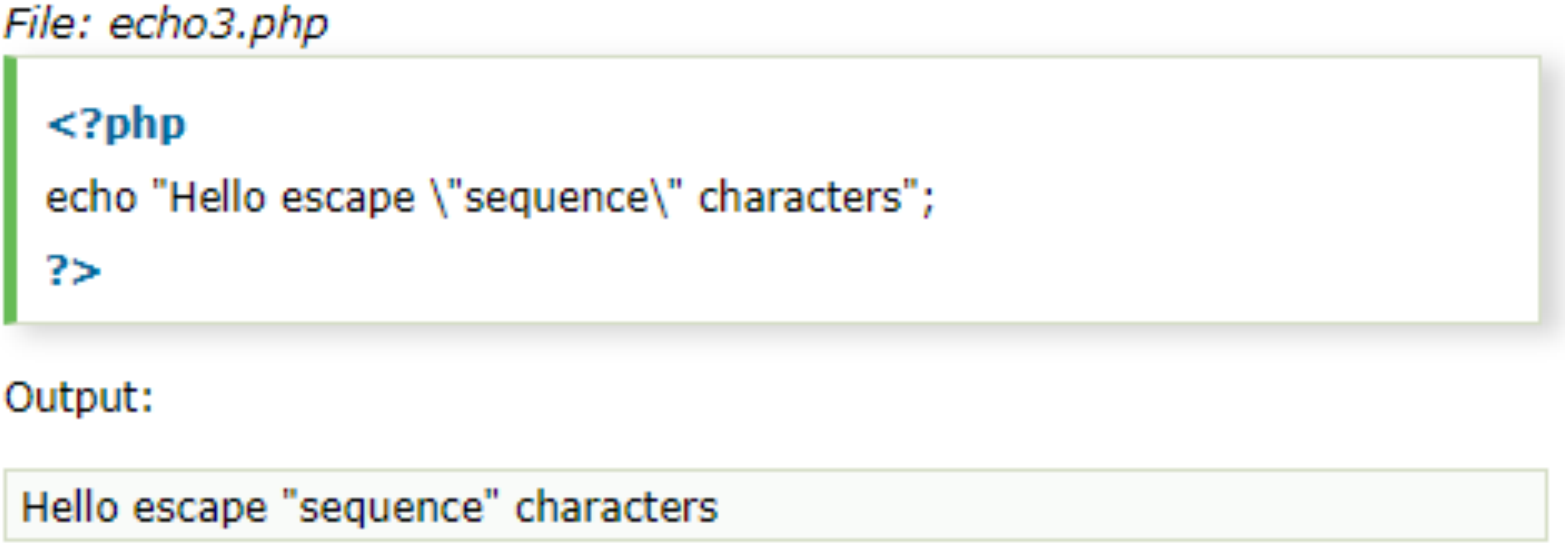


File: echo2.php

```php
<?php
echo "Hello by PHP echo
this is multi line
text printed by
PHP echo statement
";
?>
```

Output:

```
Hello by PHP echo this is multi line text printed by PHP echo statement
```

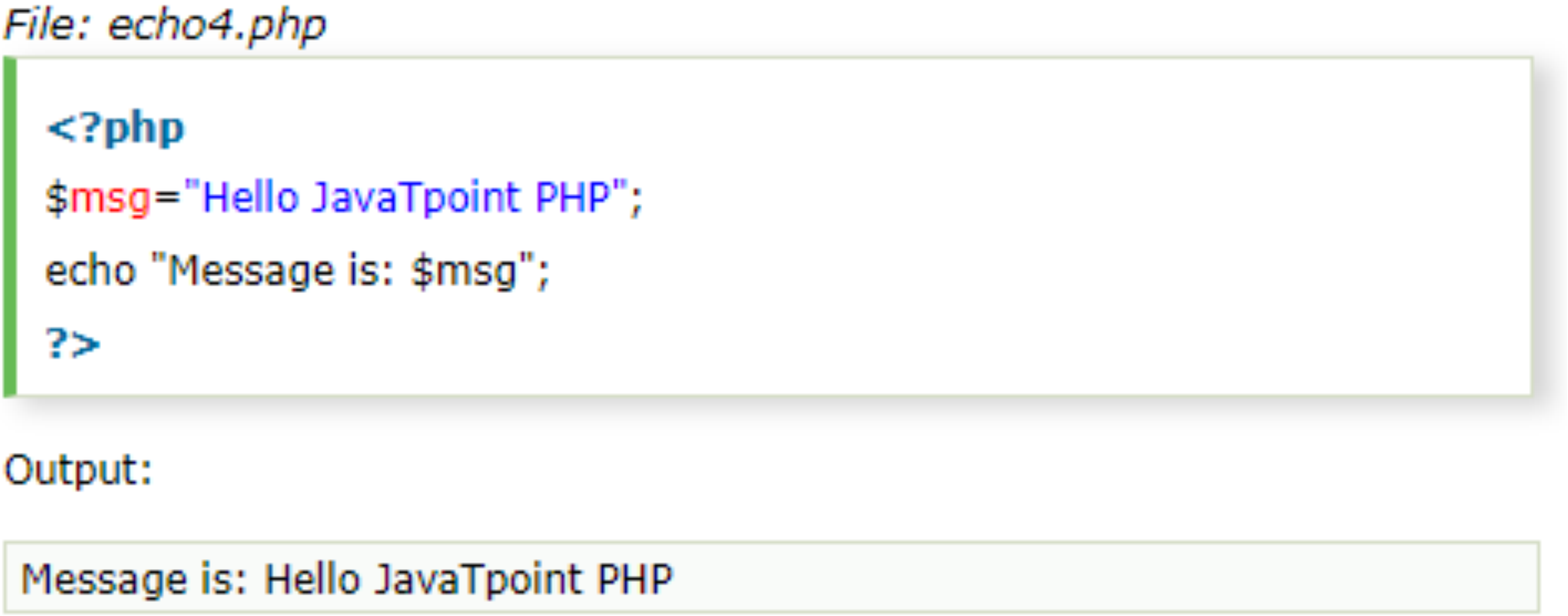


File: echo3.php

```
<?php
echo "Hello escape \"sequence\" characters";
?>
```

Output:

```
Hello escape "sequence" characters
```



File: echo4.php

```
<?php
$msg="Hello JavaTpoint PHP";
echo "Message is: $msg";
?>
```

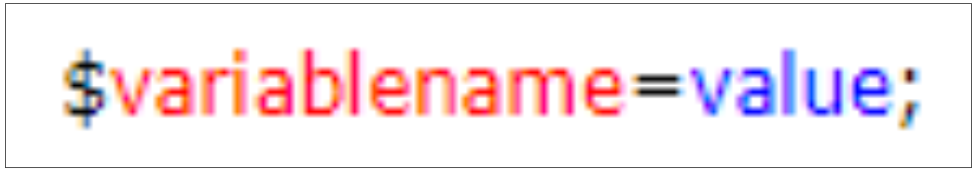
Output:

```
Message is: Hello JavaTpoint PHP
```

PHP VARIABLES

PHP VARIABLES

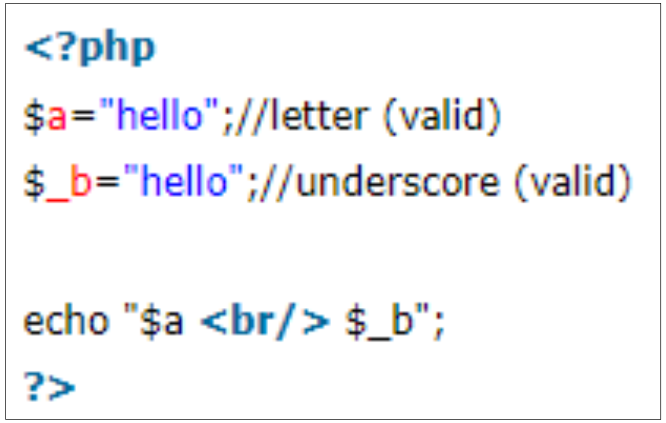
- A variable in PHP is a name of memory location that holds data. A variable is a temporary storage that is used to store data temporarily.
- In PHP, a variable is declared using $ sign followed by variable name.
- Syntax of declaring a variable in PHP is given below:



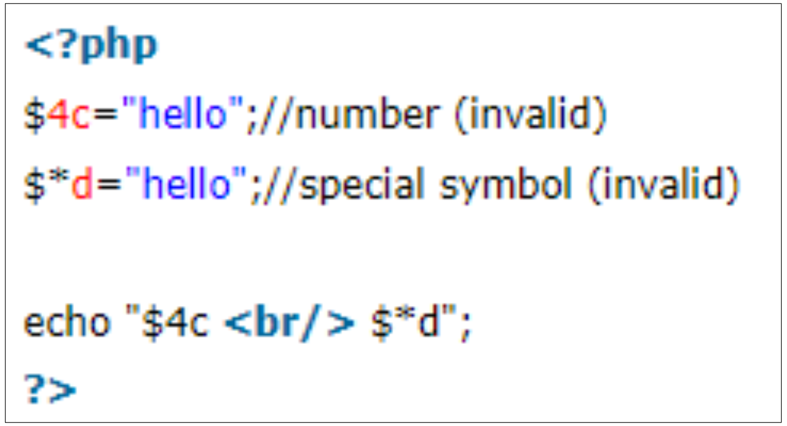
```
$variablename=value;
```

PHP VARIABLE: RULES

- PHP variables must start with letter or underscore only.
- PHP variable can't be start with numbers and special symbols.

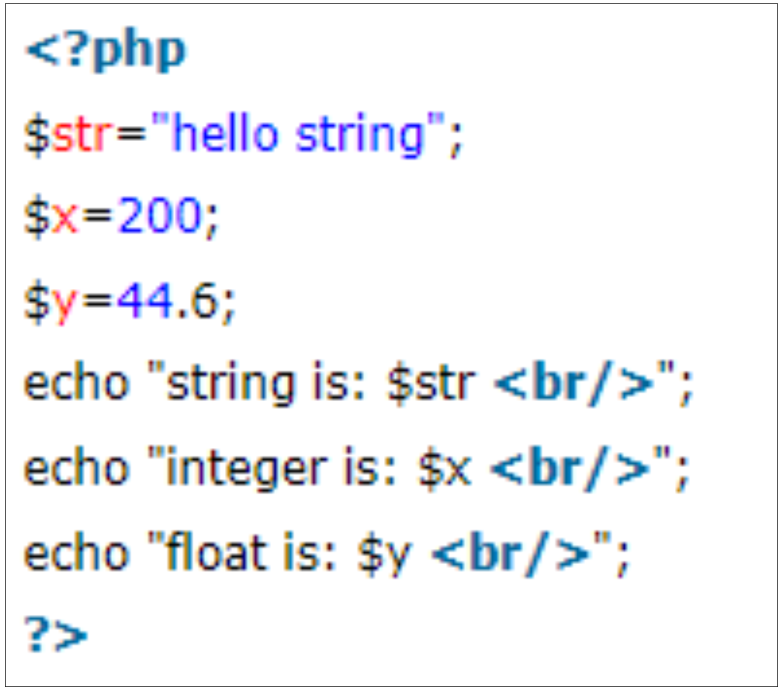


```php
<?php
$a="hello";//letter (valid)
$_b="hello";//underscore (valid)

echo "$a <br/> $_b";
?>
```



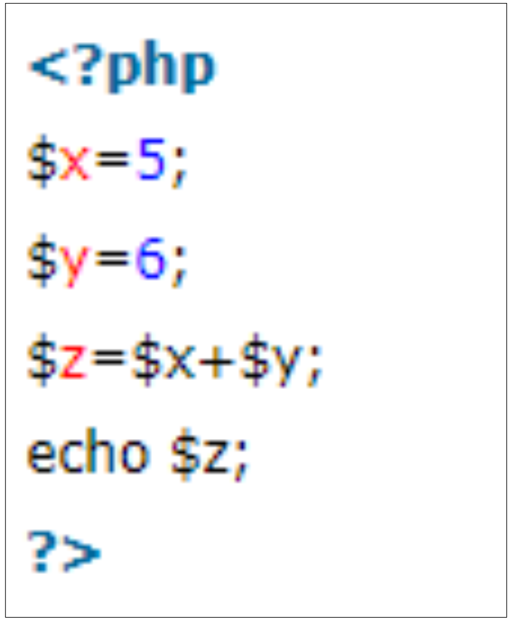
```php
<?php
$4c="hello";//number (invalid)
$*d="hello";//special symbol (invalid)

echo "$4c <br/> $*d";
?>
```

DECLARING STRING, INTEGER AND FLOAT



```
<?php
$str="hello string";
$x=200;
$y=44.6;
echo "string is: $str <br/>";
echo "integer is: $x <br/>";
echo "float is: $y <br/>";
?>
```

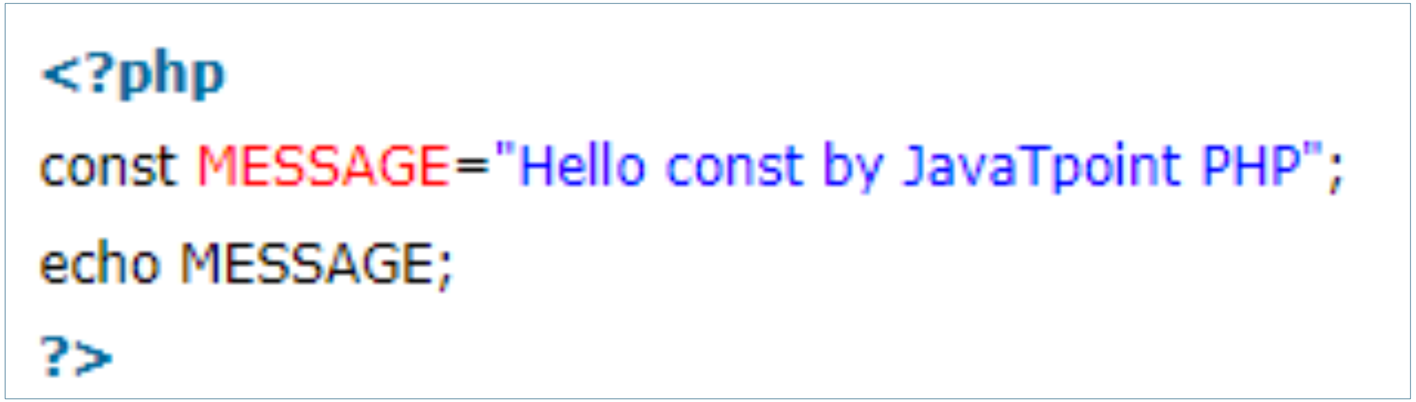
SUM OF TWO VARIABLES



```
<?php
$x=5;
$y=6;
$z=$x+$y;
echo $z;
?>
```

PHP CONSTANT: CONST KEYWORD

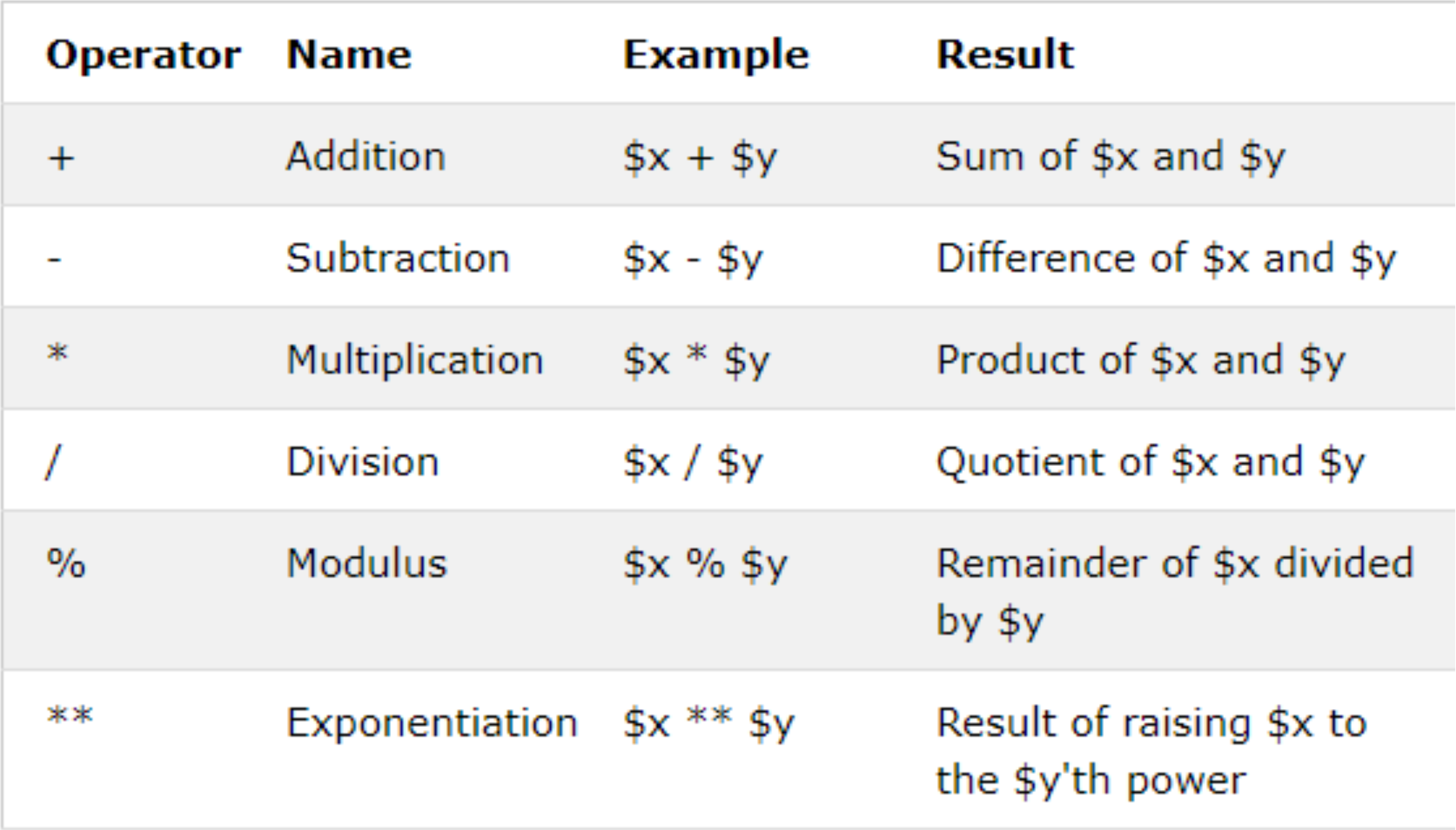
- The const keyword defines constants at compile time. It is a language construct not a function.
- It is bit faster than define().
- It is always case sensitive.



```
<?php
const MESSAGE="Hello const by JavaTpoint PHP";
echo MESSAGE;
?>
```

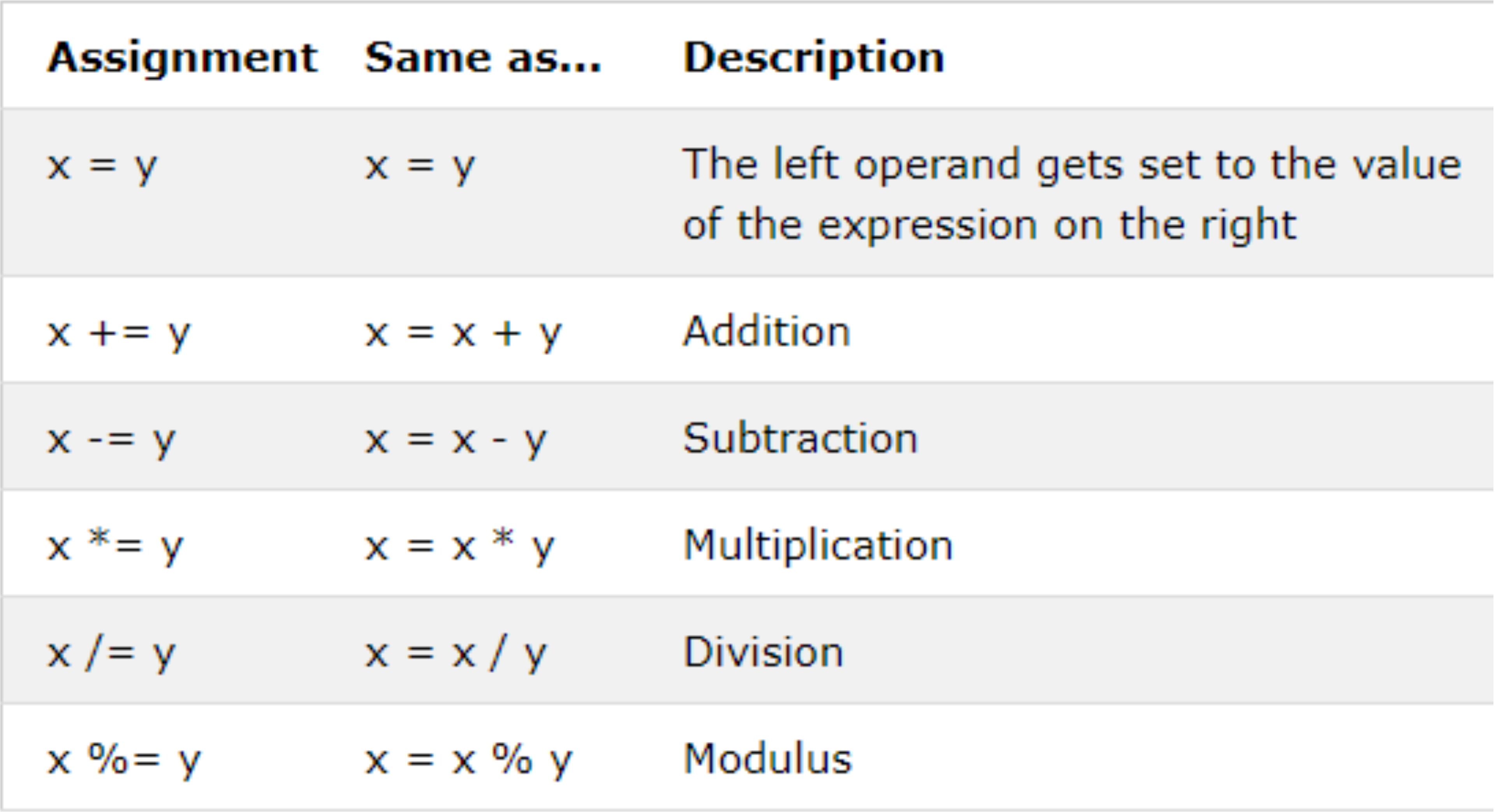
PHP OPERATORS

PHP ARITHMETIC OPERATORS



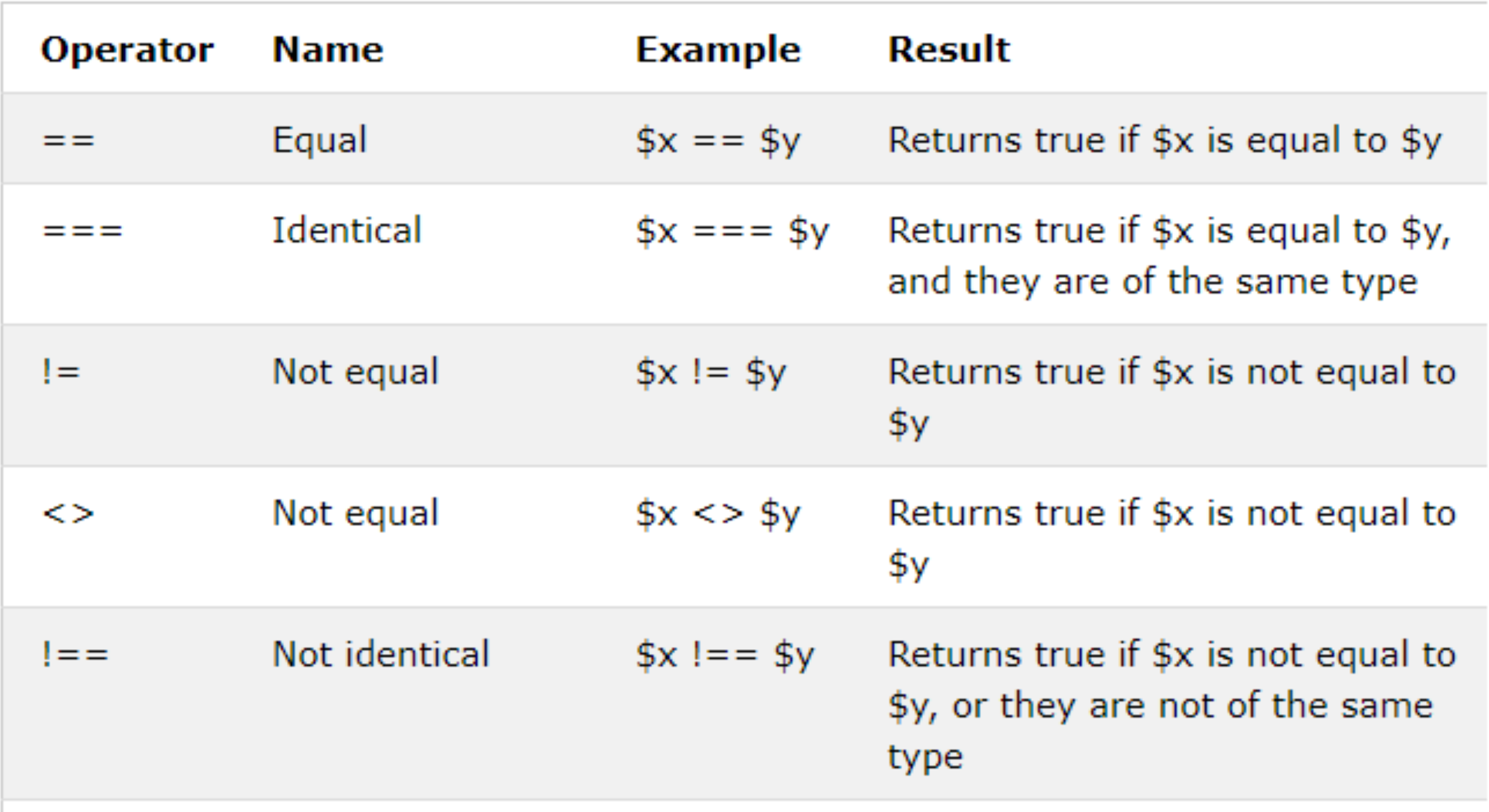
Operator	Name	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y
*	Multiplication	$x * $y	Product of $x and $y
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $y
**	Exponentiation	$x ** $y	Result of raising $x to the $y'th power

PHP ASSIGNMENT OPERATORS



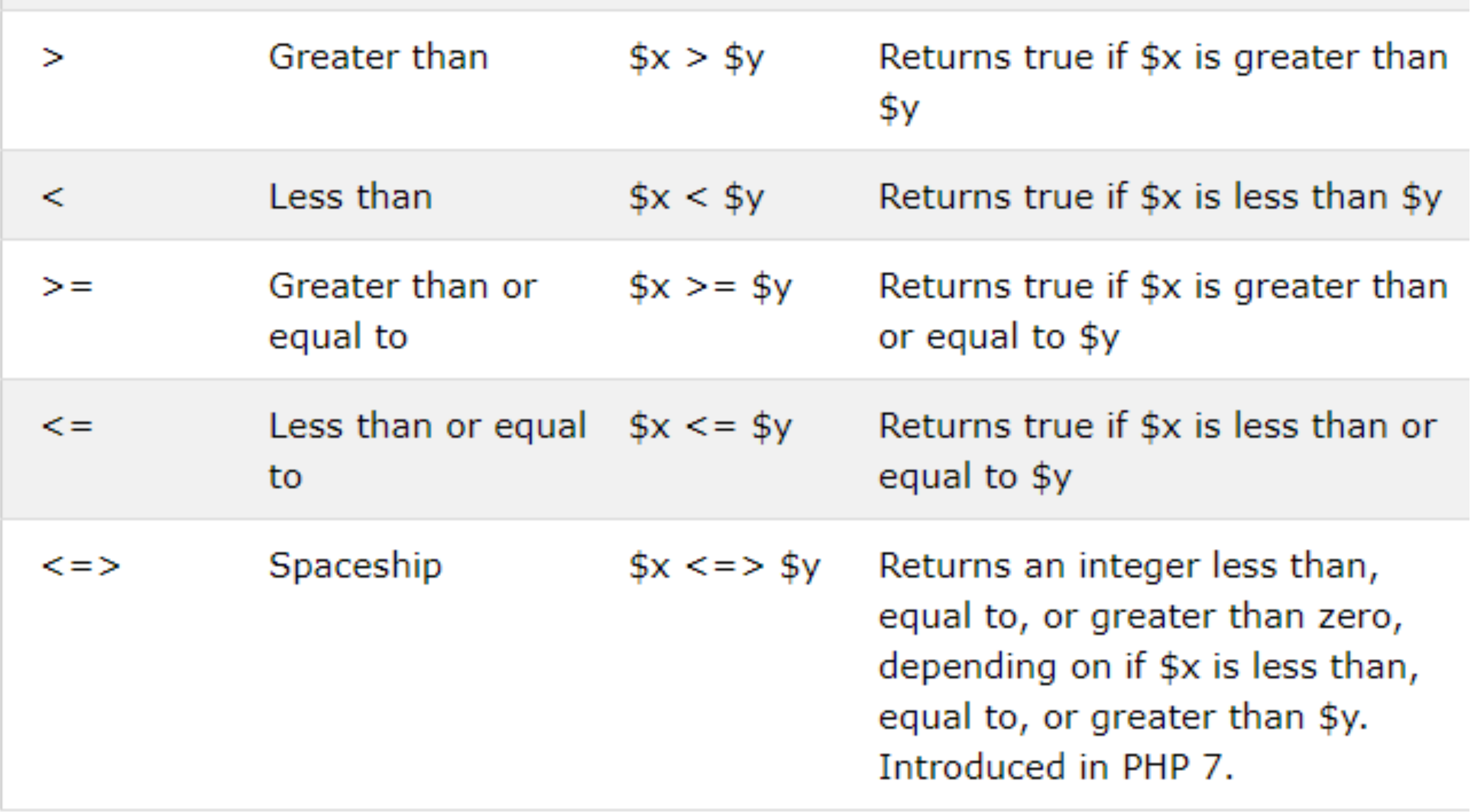
Assignment	Same as...	Description
x = y	x = y	The left operand gets set to the value of the expression on the right
x += y	x = x + y	Addition
x -= y	x = x - y	Subtraction
x *= y	x = x * y	Multiplication
x /= y	x = x / y	Division
x %= y	x = x % y	Modulus

PHP COMPARISON OPERATORS



Operator	Name	Example	Result
==	Equal	$x == $y	Returns true if $x is equal to $y
===	Identical	$x === $y	Returns true if $x is equal to $y, and they are of the same type
!=	Not equal	$x != $y	Returns true if $x is not equal to $y
<>	Not equal	$x <> $y	Returns true if $x is not equal to $y
!==	Not identical	$x !== $y	Returns true if $x is not equal to $y, or they are not of the same type

PHP COMPARISON OPERATORS



>	Greater than	$x > $y	Returns true if $x is greater than $y
<	Less than	$x < $y	Returns true if $x is less than $y
>=	Greater than or equal to	$x >= $y	Returns true if $x is greater than or equal to $y
<=	Less than or equal to	$x <= $y	Returns true if $x is less than or equal to $y
<=>	Spaceship	$x <=> $y	Returns an integer less than, equal to, or greater than zero, depending on if $x is less than, equal to, or greater than $y. Introduced in PHP 7.