PHP ARRAY FUNCTIONS

Array Length

- The ‘Count()’ function returns the number of elements in the array.

```
<?php
$items = ["a", "b", "c"];
echo count($items);
?>
```

Outputs: 3

Adding Elements to Arrays

- Adding elements in a PHP array can be done in 2 ways :

✓**Using array_push ()** : inserts the element at the end of the array.

✓**Using array_unshift() :** inserts the element at the beginning of the array.

array_push()

- The array_push() function inserts one or more elements to the end of an array.
- array_push(array, value1, value2, ...)

```
<?php
$a=array("red","green");
array_push($a,"blue","yellow");
print_r($a);
?>
```

OUTPUT:

Array ([0] => red [1] => green [2] => blue [3] => green)

array_unshift()

- The array_unshift() adds one or more elements to the beginning of an array.

```
<?php
$queue = ["orange", "banana"];
array_unshift($queue, "apple", "raspberry");
print_r($queue);
?>
```

Outputs: Array ([0] => apple [1] => raspberry [2] => orange [3] => banana)

Removing Elements from Arrays

- Removing elements in a PHP array can be done in 2 ways :

✓**Using array_pop() :** removes and returns the last element of the array.

✓**Using array_shift():** removes and returns the first element of the array.

array_pop()

- The array_pop() function deletes the last element of an array.

  array_pop(array)

```
<?php
$a=array("red","green","blue");
array_pop($a);
print_r($a);
?>
```

OUTPUT:

Array ([0] => red [1] => green)

Using array_shift()

```
<?php
$queue = ["orange", "banana", "apple"];
$fruit = array_shift($queue);
echo $fruit;
print_r($queue);
?>
```

Outputs: orange

Outputs: Array ([0] => banana [1] => apple)

array_search()

- The array_search() function search an array for a value and returns the key.
- array_search(value, array, strict)

```
<?php
$a=array("a"=>"red","b"=>"green","c"=>"blue");
echo array_search("red",$a);
?>
```

OUTPUT:

a

array_search()

```
<?php
$a=array("a"=>"1","b"=>1,"c"=>"1");
echo array_search(1,$a,true);
?>
```

OUTPUT:

b

//true is used for strict comparison

array_count_values()

- The array_count_values() function is used to count the frequency of all the values in an array.
- array_count_values(array)

```
<?php
$a=array("Block 33","Block 34","Block 34","Block 36","Block 36");
print_r(array_count_values($a));
?>
```

OUTPUT:

Array ([Block 33] => 1 [Block 34] => 2 [Block 36] => 2)

array_merge()

- The array_merge() function merges one or more arrays into one array.
- array_merge(array1, array2, array3, ...)

```
<?php
$a1=array("a"=>"red","b"=>"green");
$a2=array("c"=>"blue","b"=>"yellow");
$a3=array("c"=>"orange","b"=>"magenta");
print_r(array_merge($a1,$a2,$a3));
?>
```

OUTPUT:

Array ([a] => red [b] => magenta [c] => orange)

array_merge()

```
<?php
$a1=array("red","green", "blue");
$a2=array("blue","yellow");
print_r(array_merge($a1,$a2));
?>
```

OUTPUT:

Array ([0] => red [1] => green [2] => blue [3] => blue [4] => yellow)

array_combine()

- The array_combine() function creates an array by using the elements from one "keys" array and one "values" array.

- array_combine(keys, values)

```
<?php
$name=array("Manoj","Rahul","Aneesh");
$marks=array("75","89","44");
$c=array_combine($name,$marks);
print_r($c);
?>
```

OUTPUT:

Array ([Manoj] => 75 [Rahul] => 89 [Aneesh] => 44)

array_chunk()

- The array_chunk() function splits an array into chunks of new arrays.
- array_chunk(array, size, preserve_key)

```
<?php
$courses=array("PHP","Laravel","Node
js","HTML","CSS","ASP.NET");
print_r(array_chunk($courses,2));
?>
```

OUTPUT:

Array ([0] => Array ([0] => PHP [1] => Laravel) [1] => Array ([0] => Node js [1] => HTML) [2] => Array ([0] => CSS [1] => ASP.NET))

array_chunk()

```
<?php
$courses=array("a"=>"PHP","b"=>"Laravel","c"=>"Node js","d"=>"HTML","e"=>"CSS","f"=>"ASP.NET");
print_r(array_chunk($courses,2));
?>
```

OUTPUT:

```
Array ( [0] => Array ( [0] => PHP [1] => Laravel ) [1] => Array ( [0] => Node js [1] => HTML ) [2] => Array ( [0] => CSS [1] => ASP.NET ) )
```

array_chunk()

```
<?php
$courses=array("a"=>"PHP","b"=>"Laravel","c"=>"Node js","d"=>"HTML","e"=>"CSS","f"=>"ASP.NET");
print_r(array_chunk($courses,2, true));
?>
```

OUTPUT:

```
Array ( [0] => Array ( [a] => PHP [b] => Laravel ) [1] => Array ( [c] => Node js [d] => HTML ) [2] => Array ( [e] => CSS [f] => ASP.NET ) )
```

array_diff()

- The array_diff() function compares the values of two (or more) arrays, and returns the differences.

- This function compares the values of two (or more) arrays, and return an array that contains the entries from array1 that are not present in array2 or array3, etc.

- array_diff(array1, array2, array3, ...)

array_diff()

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yello
w");
$a2=array("e"=>"red","f"=>"green","g"=>"blue");
$a3=array("h"=>"magenta","i"=>"seagreen");
$result=array_diff($a1,$a2);
print_r($result);
?>
```

OUTPUT:

Array ([d] => yellow)

array_flip()

- The array_flip() function flips/exchanges all keys with their associated values in an array.
- array_flip(array)

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yello
w");
$result=array_flip($a1);
print_r($result);
?>
```

OUTPUT:

Array ([red] => a [green] => b [blue] => c [yellow] => d)

array_flip()

```
<?php
$a1=array("red","green","blue","yellow");
$result=array_flip($a1);
print_r($result);
?>
```

OUTPUT:

Array ([red] => 0 [green] => 1 [blue] => 2 [yellow] => 3)

array_intersect()

- The array_intersect() function compares the values of two (or more) arrays, and returns the matches.
- array_intersect(array1, array2, array3, ...)

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("e"=>"red","f"=>"green","g"=>"blue");
$a3=array("red","blue");
$result=array_intersect($a1,$a2,$a3);
print_r($result);
?>
```

OUTPUT:

Array ([a] => red [c] => blue)

array_reverse()

- The array_reverse() function returns an array in the reverse order.
- array_reverse(array, preserve)

```
<?php
$a=array("a"=>"Volvo","b"=>"BMW","c"=>"Toyota");
print_r(array_reverse($a));
?>
```

OUTPUT:

Array ([c] => Toyota [b] => BMW [a] => Volvo)

array_reverse()

- Pass true value to preserve the key

```
<?php
$a=array("Volvo","BMW","Toyota");
print_r(array_reverse($a, true));
?>
```

OUTPUT:

Array ([2] => Toyota [1] => BMW [0] => Volvo)

array_slice()

- The array_slice() function returns selected parts of an array.
- array_slice(array, start, length, preserve)

```
<?php
$a=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow","e"=>"brow
n");
print_r(array_slice($a,1,2));

echo "<br>";

$a=array("red","green","blue","yellow","brown");
print_r(array_slice($a,1,2,true));
?>
```

OUTPUT:

Array ([b] => green [c] => blue)

Array ([1] => green [2] => blue)

array_column()

- The array_column() function returns the values from a single column in the input array.
- array_column(array, column_key, index_key)

```
<?php

$result = array(
    array('name'=>'Manoj','cgpa'=>6.7,'status'=>'pass'),
    array('name'=>"Shalini",'cgpa'=>9.8,'status'=>'pass'),
    array('name'=>'Mani','cgpa'=>3.2,'status'=>'fail')
    );

$name = array_column($result, 'name');
print_r($name);
?>
```

OUTPUT:

Array ([0] => Manoj [1] => Shalini [2] => Mani)

array_column()

```
<?php

$result = array(
    array('name'=>'Manoj','cgpa'=>6.7,'status'=>'pass'),
    array('name'=>"Shalini",'cgpa'=>9.8,'status'=>'pass'),
    array('name'=>'Mani','cgpa'=>3.2,'status'=>'fail')
    );
    $names = array_column($result, 'status', 'name');
    print_r($names);
?>
```

Array ([Manoj] => pass [Shalini] => pass [Mani] => fail)

array_sum()

- returns the sum of all the values in an array.

```
<?php
$array = [1, 2, 3, 4];
$sum = array_sum($array);
echo $sum;
?>
```

Outputs: 10

array_product()

- returns the product of all the values in an array.

```
<?php
$array = [1, 2, 3, 4];
$product = array_product($array);
echo $product;
?>
```

Outputs: 24

in_array()

- checks if a value exists in the array.

```
<?php
$array = ["a", "b", "c"];
$exists = in_array("b", $array);
echo $exists ? "Exists" : "Does not exist";
?>
```

Outputs: Exists

array_key_exists()

- checks if a specific key exists in the array.

```
<?php
$array = ["a" => 1, "b" => 2, "c" => 3];
$keyExists = array_key_exists("b", $array);
echo $keyExists ? "Key exists" : "Key does not exist";
?>
```

Outputs: Key exists