

សិក្សាពី៖

សេរីទិន្នន័យ

& Array

រៀបរៀងដោយ៖ ក្រុមទី១



១. តើសេរីទិន្នន័យក្នុង PHP ជាអ្វី

Array គឺជាអថេរស្មុគស្មាញដែលអនុញ្ញាតឲ្យយើងរក្សាទុកតម្លៃច្រើនជាងមួយឬក្រុមនៃតម្លៃក្រោមឈ្មោះអថេរតែមួយ។ ការរក្សាទុកពណ៌ម្តងក្នុងមួយអថេរអាចមើលខាងក្រោម៖

ក្រោម៖

```
<?php  
    $color1 = "Red";  
    $color2 = "Green";  
    $color3 = "Blue";  
?>
```



២.ប្រភេទនៃ អារេ ក្នុង PHP – Types of Array in PHP

❖ យើងអាចបង្កើត Array បានបីយ៉ាង។ ខាងក្រោមគឺជាប្រភេទនៃ Array ៖

- Indexed array _ An array with a numeric key.
- Associative array _ An array where each key has its own specific value.
- Multidimensional array _ An array containing one or more arrays within itself.



២.១ Indexed Arrays

❖ **Indexed Arrays:** indexed ឬជាលេខរក្សាទុកធាតុអាចនីមួយៗជាមួយនឹងសន្ទស្សន៍ជាលេខ។តាម គំរូ ខាង ក្រោម បង្ហាញ ពី វិធី ពីរ យ៉ាង នៃ ការ បង្កើត array ដែល indexed វិធី ងាយ ស្រួល បំផុត គឺ :

```
<?php
// Define an indexed array
$colors = array("Red", "Green", "Blue");
?>
```

```
<?php
$colors[0] = "Red";
$colors[1] = "Green";
$colors[2] = "Blue";
?>
```



២.២ Associative Arrays

❖ **Associative Arrays** :នៅក្នុងអារេដែលជាប់ទំនាក់ទំនងគ្នា នឹង keyដែលត្រូវបានកំណត់ទៅតម្លៃអាចត្រូវបានកំណត់ដោយអ្នកប្រើ defined string។នៅក្នុងឧទាហរណ៍ខាងក្រោមនេះ array ប្រើគន្លឹះជំនួសឱ្យលេខnumbers :

```
<?php
// Define an associative array
$ages = array("Peter"=>22, "Clark"=>32, "John"=>28);
?>
```



២.៣ Multidimensional Arrays

❖ **Multidimensional Arrays** : គឺជាប្រភេទនៃ Array ដែលរក្សាទុក Array មួយផ្សេងទៀត នៅ index នីមួយៗជំនួសឱ្យធាតុ។ ម្យ៉ាងវិញទៀត កំណត់អាសយដ្ឋានត្រឹមត្រូវជាអាននៃអាន។ ឧទាហរណ៍មួយនៃ array Multidimensional នឹងមើលទៅដូចជា៖



<?php

```
// Define a multidimensional array
$contacts = array(
    array(
        "name" => "Peter Parker",
        "email" => "peterparker@mail.com",
    ),
    array(
        "name" => "Clark Kent",
        "email" => "clarkkent@mail.com",
    ),
    array(
        "name" => "Harry Potter",
        "email" => "harrypotter@mail.com",
    )
);
// Access nested value
echo "Peter Parker's Email-id is: " .
$contacts[0]["email"];
```

?>

៣ . Viewing Array Structure and Values

❖ **Viewing Array Structure and Values :** អ្នក អាច មើល ឃើញ រចនា សម្ព័ន្ធ និង គុណ តម្លៃ របស់ array ណាមួយ ដោយ ប្រើ សេចក្តី ថ្លែងការណ៍ មួយ ក្នុងចំណោម សេចក្តី ថ្លែងការណ៍ ពីរ — var _dump () ឬ print _ r() ។ សូម ពិចារណា អំពី គំរូ ខាង ក្រោម នេះ ៖

```
<?php
```

```
// Define array  
$cities = array("London", "Paris", "New York");  
// Display the cities array  
print_r($cities);
```

```
?>
```



៤ . អនុគមន៍ នៃអារេ - PHP Array Functions

នេះ ជា បញ្ជី ពេញលេញ នៃ មុខងារ array ដែល ជា របស់ PHP 7 ចុង ក្រោយ បំផុត។ ទាំងនេះ មុខងារគឺ ជាផ្នែកនៃស្នូល PHP ដូច្នេះអ្នកអាចប្រើវានៅក្នុងស្ត្រីបរបស់អ្នកដោយគ្មាន ការដំឡើងបន្ថែម៖

- `Array_change_key_case ()` : ផ្លាស់ប្តូរករណីនៃ Key ទាំងអស់ក្នុង array ដើម្បីបន្ទាប caseឬពិលើ។

```
<?php
// Sample array
$persons = array("Harry"=>22, "Clark"=>32,
"John"=>28);

// Changing keys to lowercase
print _r(array _ change _key _ case($persons));
?>
```



៤ . អនុគមន៍ នៃអារេ - PHP Array Functions

- `Array_chunk ()` : បំបែក array ទៅជាកំណាត់នៅក្នុង array ។

```
<?php
// Sample array
$colors = array("red", "green", "blue", "orange", "yellow", "black");

// Split colors array into chunks
print_r(array_chunk($colors, 2));
?>
```



- `Array_column()` : ត្រឡប់ តម្លៃ ពី ជួរ ឈរ តែ មួយ ក្នុង បញ្ចូល array។

EX:

```
<?php
// Sample array
$movies = array(
    array(
        "id" => "1",
        "name" => "Titanic",
        "genre" => "Drama",
    ),
    array(
        "id" => "2",
        "name" => "Justice League",
        "genre" => "Action",
    ),
    array(
        "id" => "3",
        "name" => "Joker",
        "genre" => "Thriller",
    )
);

// Getting the column of names
$names = array_column($movies, "name", "id");
print_r ($names);
?>
```

- `Array_combine()` : បង្កើត array ដោយ ប្រើ array មួយ សម្រាប់ គន្លឹះ និង មួយ ទៀត សម្រាប់ តម្លៃ របស់

វា ។ Ex: `<?php`

```
// Sample arrays
$array1 = array("a", "b", "c", "d");
$array2 = array("apple", "ball", "cat",
"dog");

// Combining both arrays
print_r (array_combine($array1, $array2));
?>
```

- `Array_count_values()` : រាប់តម្លៃផ្សេងគ្នាទាំងអស់នៃអារេមួយ។

Ex: `<?php`

```
// Sample array
$letters = array("a", "b", "a", "c", "b", "a");

// Counting array values
Print_r (array_count_values($letters));
?>
```

- `Array_diff()` : ប្រៀបធៀបតម្លៃអាន ហើយ return ភាពខុសគ្នា។

Ex:

```
<?php
// Sample arrays
$array1 = array("apple", "ball", "cat", "dog", "elephant");
$array2 = array("alligator", "dog", "elephant", "lion", "cat");

// Computing the difference
$result = array_diff($array1, $array2);
print_r($result);
?>
```

- `Array_diff_assoc()` : ប្រៀបធៀបតម្លៃអាន ដោយមានការត្រួតពិនិត្យ all key បន្ថែម និង return ភាពខុសគ្នា។

Ex:

```
<?php
// Sample arrays
$array1 = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "dog");
$array2 = array("a"=>"apple", "banana", "papaya");

// Computing the difference
$result = array_diff_assoc($array1, $array2);
print_r($result);
?>
```

- `Array_diff_key()`: ប្រៀបធៀបគ្នាពីរ arrays ហើយត្រឡប់មកវិញនូវភាពខុសគ្នា។

Ex:

```
<?php
// Sample arrays
$array1 = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "dog");
$array2 = array("a"=>"apricot", "b"=>"banana");

// Computing the difference
$result = array_diff_key($array1, $array2);
print_r($result);
?>
```

- `Array_diff_uassoc()`: ប្រៀបធៀបតម្លៃអាវេ ជាមួយនឹងការពិនិត្យមើលkeyបន្ថែមដោយប្រើ មុខងារ ប្រៀបធៀប គន្លឹះ ដែល កំណត់ ដោយ អ្នក ប្រើ ហើយ បង្ហាញ លទ្ធផល ភាពខុសគ្នា។

Ex: `<?php`

```
// Sample arrays
```

```
$array1 = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "dog");
```

```
$array2 = array("A"=>"APPLE", "B"=>"ball", "C"=>"camel");
```

```
// Computing the difference
```

```
$result = array_diff_uassoc($array1, $array2, "strcasecmp");
```

```
print_r($result);
```

```
?>
```

- `Array_diff_ukey()` :ប្រៀបធៀប គន្លឹះ array ដោយ ប្រើ គន្លឹះ ដែល បាន កំណត់ ដោយ អ្នក ប្រើ អនុគមន៍ប្រៀបធៀប, ហើយ ត្រឡប់មកវិញភាពខុសគ្នា។ Ex: `?php`

```
// Sample arrays
$array1 = array("cat"=>2, "lion"=>5, "zebra"=>8);
$array2 = array("CAT"=>3, "FOX"=>1, "LION"=>5, "WOLF"=>7);

// Computing the difference
$result = array_diff_ukey($array1, $array2, "strcasecmp");
print_r($result);

?>
```

- `Array_fill()` :បំពេញ array ជាមួយតម្លៃ។ Ex: `<?php`

```
// Filling arrays
$array1 = array_fill(1, 5, "apple");
$array2 = array_fill(-2, 6, "banana");

// Printing the arrays
print_r($array1);
print_r($array2);

?>
```


- `Array_fill_keys()`:បំពេញអារេជាមួយតម្លៃ បញ្ជាក់keys។

Ex: `<?php`

```
// Defining keys array
$keys = array("foo", "bar", "baz");

// Filling array
$result = array_fill_keys($keys, "hello");
print_r($result);

?>
```

- `Array_filter()`:ត្រង់ធាតុនៃអារេដោយប្រើ user defined function។

Ex: `<?php`

```
// Sample array
$numbers = array("a"=>1, "b"=>2, "c"=>3, "d"=>4, "e"=>5);

// Filtering numbers array using key
$result = array_filter($numbers, function($key){
    return $key > "b";
}, ARRAY_FILTER_USE_KEY);
print_r($result);

?>
```

- `Array _ flip ()` : ត្រឡប់ ឬផ្លាស់ប្តូរkeyទាំងអស់ជាមួយនឹងតម្លៃដែលពាក់ព័ន្ធរបស់វានៅក្នុងអារេមួយ។

Ex :

```
<?php
// Defining array
$alphabets = array("a"=>"apple", "b"=>"ball", "c"=>"cat");

// Flipping alphabets array
$result = array _ flip ($alphabets);
print _r($result);
?>
```

- `Array _ intersect ()` : ប្រៀបធៀបkeyអារេ ហើយreturn matches។

Ex:

```
<?php
// Sample arrays
$array1 = array("apple", "ball", "cat", "dog", "elephant");
$array2 = array("alligator", "dog", "elephant", "lion", "cat");

// Computing the intersection
$result = array _ intersect ($array1, $array2);
print _r($result);
?>
```

- `Array _intersect _assoc ()` : ប្រៀបធៀបតម្លៃ array ជាមួយនឹងការពិនិត្យមើលkeyបន្ថែមដោយប្រើមុខងារ។

Ex : <?php
 // Sample arrays
 \$array1 = array(0, 1, 2, 5, 7);
 \$array2 = array("00", "1", 2, "05", 8, 7);

 // Computing the intersection
 \$result = array _intersect _assoc(\$array1, \$array2);
 print _r(\$result);
 ?>

- `Array _intersect _key ()`: ប្រៀបធៀបkeyដែលកំណត់ដោយអ្នកប្រើប្រាស់ ហើយreturn matches។

Ex: <?php
 // Sample arrays
 \$array1 = array(1, 2, 5, 7, 10);
 \$array2 = array(0, "1"=>3, "x"=>8, "4"=>13);

 // Computing the intersection
 \$result = array _intersect _key (\$array1, \$array2);
 print _r(\$result);
 ?>

- `Array _intersect _uassoc ()` : ប្រៀបធៀបkey arrayដោយប្រើមុខងារ។

Ex: `<?php`
`// Sample arrays`
`$array1 = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "dog");`
`$array2 = array("A"=>"ant", "B"=>"ball", "C"=>"camel");`
`$array3 = array("a"=>"airplane", "b"=>"ball");`

`// Computing the intersection`
`$result = array _intersect _uassoc ($array1, $array2, $array3, "strcasecmp");`
`print _r($result);`

`Array _?>intersect _ukey()` : ប្រៀបធៀបkeyដែលកំណត់ដោយអ្នកប្រើប្រាស់ ហើយត្រឡប់ការផ្ដផ្គង។

Ex: `<?php`
`// Sample arrays`
`$array1 = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "dog");`
`$array2 = array("A"=>"apricot", "b"=>"banana");`
`$array3 = array("a"=>"alligator", "b"=>"balloon");`

`// Computing the difference`
`$result = array _intersect _ukey ($array1, $array2, $array3, "strcasecmp");`
`print _r($result);`

`?>`

- `Array _ key _ exists()`:ពិនិត្យមើលkeyដែលបានបញ្ជាក់នៅក្នុងarray។

Ex:

```
<?php
// Sample array
$lang = array("en"=>"English", "fr"=>"French", "ar"=>"Arabic");

// Test if key exists in the array
if(array _ key _ exists("fr", $lang)){
    echo "Key exists!";
} else{
    echo "Key does not exist!";
}
?>
```

- `Array _ key _ first()` :ទទួលយកkeyដំបូងនៃអារ។

Ex:

```
<?php
// Sample array
$persons = array("Harry"=>18, "Clark"=>32, "John"=>24);

// Getting the first key from the persons array
echo array _ key _ first($persons); // Prints: Harry
?>
```

- `Array_key_last()`: ទទួលយក key ចុងក្រោយនៃ array ។

Ex:

```
<?php
// Sample array
$persons = array("Harry"=>18, "Clark"=>32, "John"=>24);

// Getting the last key from the persons array
echo array_key_last($persons); // Prints: John
?>
```

- `Array_keys()`: Return key ទាំងអស់ ឬសំណុំនៃ key នៃ array មួយ។

Ex:

```
<?php
// Sample array
$persons = array("Harry"=>18, "Clark"=>"32", "John"=>24, "Peter"=>32);

// Getting all the keys from the persons array
print_r(array_keys($persons));
?>
```

- `Array_map()`: បញ្ជូនធាតុនៃarrayដែលបានផ្តល់ឱ្យទៅ user defined function ដែលអាចប្រើវាដើម្បីreturnតម្លៃថ្មី។

Ex:

```
<?php
// Sample arrays
$fruits = array("apple", "banana", "orange", "mango");

// Applying callback function to each element
$result = array_map("strtoupper", $fruits);
print_r($result);
?>
```

- `Array_merge()`: បញ្ចូល array មួយ ឬច្រើនទៅក្នុង array មួយ។

Ex:

```
<?php
// Sample arrays
$array1 = array("red", "green", "blue");
$array2 = array("apple", "banana");

// Merging the two indexed array
$result = array_merge($array1, $array2);
print_r($result);
?>
```

- `Array _ merge _ recursive()`: បញ្ចូល array មួយឬច្រើនទៅក្នុង array មួយដដែលៗ។

Ex:

```
<?php
// Sample arrays
$array1 = array("fruits"=>array("a"=>"apple"), 5);
$array2 = array(10, "fruits"=>array("a"=>"apricot", "banana"));

// Merging the two arrays
$result = array _ merge _ recursive($array1, $array2);
print _r($result);
?>
```

- `Array _ multisort ()`: តម្រៀប array ច្រើន ឬច្រើនវិមាត្រ។

Ex:

```
<?php
// Sample arrays
$array1 = array(2, 7, 10, 5);
$array2 = array(4, 3, 1, 2);

// Sorting multiple arrays
array _ multisort ($array1, $array2);
print _r($array1);
print _r($array2);
?>
```


- `Array_pad()`: បញ្ចូលចំនួនធាតុដែលបានបញ្ជាក់ដោយ item ដោយតម្លៃដែលបានបញ្ជាក់ទៅ array មួយ។

Ex:

```
<?php
// Sample array
$numbers = array(5, 10, 15);

// Padding numbers array
print _r(array _ pad($numbers, 5, 0));
?>
```

- `Array_pop()`: លុបធាតុចុងក្រោយនៃ array មួយ ហើយ return តម្លៃនៃធាតុដែលបានដកចេញ។

Ex:

```
<?php
// Sample array
$cities = array("London", "Paris", "New York", "Sydney");

// Remove and get the last value from array
echo array _ pop($cities); // Prints: Sydney
print _r($cities);
?>
```

- `Array_product()`: គណនាលទ្ធផលនៃតម្លៃក្នុង array មួយ។

Ex:

```
<?php
// Sample array
$numbers = array(2, 5, 8, 10, 15);

// Getting the product of array values
echo array_product($numbers); // Prints: 12000
?>
```

- `Array_push()`: បញ្ចូលធាតុមួយ ឬច្រើនទៅចុងបញ្ចប់នៃ array មួយ។

Ex:

```
<?php
// Sample array
$colors = array("red", "green", "blue");

// Pushing values to the array
array_push($colors, "yellow", "orange");
print_r($colors);
?>
```

- `Array _ rand()`: Return key មួយ ឬច្រើនពី array មួយ។

Ex:

```
<?php
// Sample array
$colors = array("red", "green", "blue", "yellow", "orange");

// Getting two random keys from the colors array
$rand _ keys = array _ rand($colors , 2);
print _ r($rand _ keys);

// Printing corresponding values from colors array
echo $colors [$rand _ keys[0]] . "<br>";
echo $colors [$rand _ keys[1]];
?>
```

- `Array_reduce()` កាត់បន្ថយ array ទៅជាតម្លៃតែមួយដោយប្រើមុខងារ callback ដែលកំណត់ដោយអ្នកប្រើប្រាស់។

Ex:

```
<?php
// Defining a callback function
function sum($carry, $item){
    $carry += $item;
    return $carry;
}

// Sample array
$numbers = array(1, 2, 3, 4, 5);
var_dump(array_reduce($numbers, "sum")); // int(15)
```

- `Array_replace()` : ជំនួសតម្លៃនៃ array ទីមួយជាមួយនឹងតម្លៃពី array ខាងក្រោម។

Ex:

```
<?php
// Sample arrays
$array1 = array("tea", "coffee", "chips");
$array2 = array("apple", "orange", "nuts");

// Replace the values of array1 with the
values of array2
$result = array_replace($array1, $array2);
print_r($result);
?>
```

- `Array_replace_recursive()`: ជំនួសតម្លៃនៃអារេទីមួយជាមួយនឹងតម្លៃពីអារេបន្តបន្ទាប់គ្នាឡើងវិញ។

Ex:

```
<?php
// Sample arrays
$array1 = array("pets">array("cat"), "wilds">array("wolf", "fox"));
$array2 = array("pets">array("dog", "horse"), "wilds">array("tiger"));

// Performing array replacement recursively
$result = array_replace_recursive($array1, $array2);
print_r($result);
?>
```

- `Array_reverse()`: Return arrayជាមួយធាតុក្នុងលំដាប់បញ្ជ្រាស។

Ex:

```
<?php
// Sample array
$fruits = array("apple", "banana", "orange", "mango");

// Reversing the order of the array
print_r(array_reverse($fruits));
?>
```

- `Array_search()`: ស្វែងរកarrayសម្រាប់តម្លៃដែលបានផ្តល់ឱ្យ ហើយreturn keyដែលត្រូវគ្នាប្រសិនបើជោគជ័យ។

Ex:

```
<?php
// Sample array
$alphabets = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "d"=>"dog");

// Searching array for a value
echo array_search("ball", $alphabets); // Prints: b
echo array_search("dog", $alphabets);  // Prints: d
?>
```

- `Array_shift()`: យកធាតុទីមួយចេញពី array មួយ ហើយreturnតម្លៃនៃធាតុដែលបានដកចេញ។

Ex:

```
<?php
// Sample array
$fruits = array("apple", "banana", "orange", "mango");

// Remove and get the first value from array
echo array_shift($fruits); // Prints: apple
print_r($fruits);
?>
```

- `Array_slice()`: ដក ធាតុមួយចេញពី array ។

Ex:

```
<?php
// Sample array
$fruits = array("apple", "banana", "orange", "mango", "papaya", "kiwi");

// Slicing the fruits array
$result = array_slice($fruits, 1, 3);
print_r($result);
?>
```

- `Array_splice()`: ដកធាតុមួយនៃ array ហើយជំនួសវាដោយអ្វីផ្សេងទៀត។

Ex:

```
<?php
// Sample array
$input = array("red", "green", "blue", "pink", "yellow", "black");

// Performing array splice
$result = array_splice($input, -4, -1);
print_r($result);
print_r($input);
?>
```

- `Array_sum()`: គណនាផលបូកនៃតម្លៃក្នុង array មួយ។

Ex:

```
<?php
// Sample array
$numbers = array(1, 2, 5, 7, 10);

// Getting the sum of array values
echo array_sum($numbers); // Prints: 25
```

- `Array_udiff()`: រៀបរៀបតម្លៃអានដោយប្រើ user defined ប្រៀបធៀប callback function ហើយ return ភាពខុសគ្នា។

Ex:

```
<?php
// Sample arrays
$array1 = array("a"=>"red", "b"=>"green", "c"=>"blue",
"d"=>"yellow");
$array2 = array("x"=>"black", "y"=>"BLUE", "z"=>"Red");

// Computing the difference
$result = array_udiff($array1, $array2, "strcasecmp");
print_r($result);

?>
```


- `Array_udiff_assoc()`: ប្រៀបធៀបតម្លៃអាវដោយប្រើ user defined function ប្រៀបធៀប callback function ជាមួយនឹងការពិនិត្យមើល key បន្ថែម និង return ភាពខុសគ្នា។

Ex:

```
<?php
// Sample arrays
$array1 = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "d"=>"dog");
$array2 = array("A"=>"APPLE", "b"=>"ball", "c"=>"camel");
$array3 = array("c"=>"Cat", "d"=>"DOG");

// Computing the difference
$result = array_udiff_assoc($array1, $array2, $array3, "strcasecmp");
print_r($result);
```

- `Array_udiff_uassoc()`: ប្រៀបធៀប key និងតម្លៃនៃអាវដោយប្រើ key ដែលកំណត់ដោយអ្នកប្រើប្រាស់ពីរដាច់ដោយឡែក និងប្រៀបធៀបតម្លៃ callback function ហើយ return ភាពខុសគ្នា។

Ex:

```
<?php
// Sample arrays
$array1 = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "d"=>"dog");
$array2 = array("a"=>"APPLE", "B"=>"ball", "c"=>"camel");
$array3 = array("d"=>"DOG", "e"=>"elephant");

// Computing the difference
$result = array_udiff_uassoc($array1, $array2, $array3, "strcasecmp", "strcasecmp");
print_r($result);
?>
```

- `Array _ uintersect ()`: ប្រៀបធៀបតម្លៃអារដោយប្រើ user defined function ប្រៀបធៀប callback function ហើយ return ការផ្គូផ្គង (matches) ។ Ex:

```
<?php
// Sample arrays
$array1 = array("a"=>"red", "b"=>"green", "c"=>"blue", "d"=>"yellow");
$array2 = array("x"=>"black", "y"=>"BLUE", "z"=>"Red");

// Computing the intersection
$result = array _ uintersect($array1, $array2, "strcasecmp");
print _r($result);
?>
```

- `Array _ uintersect _ assoc()`: ប្រៀបធៀបតម្លៃអារដោយប្រើ user defined function ប្រៀបធៀប callback function ជាមួយនឹងការពិនិត្យ key បន្ថែម និង return matches ។

Ex:

```
<?php
// Sample arrays
$array1 = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "d"=>"dog");
$array2 = array("A"=>"APPLE", "b"=>"BALL", "c"=>"CAMEL", "d"=>"DOG");
$array3 = array("a"=>"Apple", "b"=>"Ball", "c"=>"Cat", "d"=>"Dog");

// Computing the intersection
$result = array _ uintersect _ assoc($array1, $array2, $array3, "strcasecmp");
print _r($result);
?>
```

- `Array_uintersect_uassoc()`: ប្រៀបធៀប key និងតម្លៃនៃអារេដោយប្រើ two separate user defined key និង ប្រៀបធៀបតម្លៃ callback function ហើយនិង return matches។

Ex: `<?php`
 // Sample arrays
 \$array1 = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "d"=>"dog");
 \$array2 = array("A"=>"APPLE", "B"=>"BALL", "C"=>"CAMEL");
 \$array3 = array("a"=>"Apple", "b"=>"Banana");

 // Computing the intersection
 \$result = array_uintersect_uassoc(\$array1, \$array2, \$array3,
 "strcasecmp", "strcasecmp");
 print_r(\$result);

- `Array_unique()`: យកតម្លៃស្អាតចេញពីអារេមួយ។

Ex: `<?php`
 // Sample array
 \$numbers = array(1, 2, 4, 5, 2, 5, 7, 2, 10);

 // Removing the duplicate values from numbers array
 \$result = array_unique(\$numbers);
 print_r(\$result);
 ?>

- `Array_unshift()`:បន្ថែមធាតុមួយ ឬច្រើនទៅអានដំបូង។

Ex:

```
<?php
// Sample array
$colors = array("red", "green", "blue");

// Prepending two values to the colors array
array_unshift($colors, "yellow", "orange");
print_r($colors);
?>
```

- `Array_values()`:Returnតម្លៃទាំងអស់នៃអានមួយ។

Ex:

```
<?php
// Sample array
$alphabets = array("a"=>"apple", "b"=>"ball", "c"=>"cat", "d"=>"dog");

// Getting all the values from alphabets array
$result = array_values($alphabets);
print_r($result);
?>
```

- `Array_walk()`:អនុវត្ត user defined functionទៅធាតុនីមួយៗនៃអារេមួយ។

Ex:

```
<?php
// Defining a callback function
function myFunction($value, $key){
    echo "<p>$key for $value</p>";
}
// Sample array
$alphabets = array("a"=>"apple", "b"=>"ball", "c"=>"cat");
array_walk ($alphabets, "myFunction");
?>
```

- `Array_walk_recursive()`:អនុវត្ត user defined functionម្តងទៀតចំពោះធាតុនីមួយៗនៃអារេមួយ។

Ex:

```
<?php
// Defining a callback function
function myFunction($item, $key){
    echo "<p>$key holds $item</p>";
}
// Sample arrays
$pets = array("c" => "cat", "d" => "dog");
$animals = array("pets" => $pets, "wild" => "tiger");
array_walk_recursive($animals, "myFunction");
?>
```

- `Array ()`:បង្កើតarray។ Ex:

```
<?php
// Creating an array
$colors = array("red", "green", "blue", "yellow");
Print_r($colors);
?>
```

- `Arsort ()`:តម្រៀបអារេដែលពាក់ព័ន្ធតាមតម្លៃ ក្នុងលំដាប់បញ្ជាស ឬតាមលំដាប់ចុះ។

Ex:

```
<?php
// Sample array
$alphabets = array("b"=>"ball", "d"=>"dog", "a"=>"apple", "c"=>"cat");

// Sorting alphabets array
arsort ($alphabets);
print _r($alphabets);
?>
```

- `Asort ()`: តម្រៀប array ពាក់ព័ន្ធតាមតម្លៃនិង តាមលំដាប់ឡើង។

Ex:

```
<?php
// Sample array
$alphabets = array("b"=>"ball", "d"=>"dog", "a"=>"apple", "c"=>"cat");

// Sorting alphabets array
asort($alphabets);
print_r($alphabets);
?>
```

- `compact()`: បង្កើត array ដែលមានអថេរ និងតម្លៃរបស់វា។

Ex:

```
<?php
// Sample variables
$brand = "Apple";
$model = "iPhone";
$color = "Black";

// Creating array
$result = compact("brand", "model", "color");
print_r($result);
?>
```

- `count()`: ត្រឡប់ចំនួនធាតុនៅក្នុង array មួយ។ Ex:

```
<?php
// Sample array
$cars = array("Audi", "BMW", "Volvo",
              "Toyota");

// Display array elements count
echo count($cars);
?>
```

- `current()`: ត្រឡប់ធាតុបច្ចុប្បន្ននៅក្នុង array មួយ ។

Ex:

```
<?php
// Sample array
$colors = array("red", "green", "blue",
                "orange", "yellow", "black");

// Getting the current element
echo current($colors); // Prints: red
?>
```


- `end()`: កំណត់ទ្រនិចខាងក្នុងនៃ array ទៅធាតុចុងក្រោយរបស់វា។

Ex:

```
<?php
// Sample array
$colors = array("red", "green", "blue",
"orange", "yellow", "black");

// Getting the last element
echo end($colors); // Prints: black
?>
```

- `Extract ()`: នាំចូលអថេរទៅក្នុងតារាងនិមិត្តសញ្ញាបច្ចុប្បន្នពី array មួយ។

Ex:

```
<?php
// Sample associative array
$array = array("brand"=>"Porsche", "model"=>"911", "color"=>"blue");

// Extracting variables
extract($array);
echo "Brand: $brand, Model: $model, Color: $color";
?>
```

- `in_array()`: ពិនិត្យមើលថាតើតម្លៃមាននៅក្នុង array ។

Ex: `<?php`

```
// Sample array
$colors = array("red", "green", "blue", "orange", "yellow");
```

```
// Searching value inside colors array
```

```
if(in_array("orange", $colors)){
    echo "Match found!";
} else{
    echo "No match found!";
}
```

- `key()`: ទាញយក key ពី array មួយ។ Ex:

`<?php`

```
// Sample array
$colors = array("red", "green", "blue", "orange", "yellow", "black");
```

```
// Getting the values
```

```
echo current($colors); // Prints: red
echo end($colors);    // Prints: black
echo current($colors); // Prints: black
echo prev($colors);   // Prints: yellow
echo current($colors); // Prints: yellow
echo next($colors);   // Prints: black
echo current($colors); // Prints: black
echo reset($colors);  // Prints: red
echo current($colors); // Prints: red
```

```
// Getting the current element's key
```

```
echo key($colors);    // Prints: 0
```

`?>`

- `Krsort ()`: តម្រៀប អារេ សហការ ដោយ key ក្នុង លំដាប់ បញ្ជាស ឬ ចុះ។

Ex:

```
<?php
// Sample array
$alphabets = array("b"=>"ball", "d"=>"dog", "a"=>"apple", "c"=>"cat");

// Sorting alphabets array
krsort ($alphabets);
print _r($alphabets);
?>
```

- `Ksort ()`: តម្រៀបអារេដែលពាក់ព័ន្ធដោយkey តាមលំដាប់ឡើង។

Ex:

```
<?php
// Sample array
$alphabets = array("b"=>"ball", "d"=>"dog",
"a"=>"apple", "c"=>"cat");

// Sorting alphabets array
ksort($alphabets);
print_r($alphabets);
?>
```

- `list()`: កំណត់អថេរដូចជាប្រសិនបើវាជាអារ។

Ex: `<?php`
`// Sample array`
`$phone = array("Apple", "iPhone", "128GB");`

`// Listing all the variables`
`list($brand, $model, $rom) = $phone;`
`echo "This is an $brand $model with $rom internal storage.";`

- `natcasesort ()`: តម្រៀបអារដោយប្រើក្បួនដោះស្រាយ "លំដាប់ធម្មជាតិ" ដែលមិនប្រកាន់អក្សរតូចធំ។

Ex: `<?php`
`// Sample array`
`$images = array("IMG5.png", "img10.png", "IMG2.png", "img1.png");`

`// Standard sorting`
`sort($images);`
`print _r($images);`

`// Natural order sorting`
`natcasesort ($images);`
`print _r($images);`
`?>`

- `Natsort ()`: តម្រៀបអារេដោយប្រើ "ក្បួនដោះស្រាយលំដាប់ធម្មជាតិ។

Ex: `<?php`
 // Sample array
 \$images = array("img5.png", "img10.png", "img2.png", "img1.png");

 // Standard sorting
 sort(\$images);
 print_r(\$images);

 // Natural order sorting
 natsort (\$images);
 print_r(\$images);
 ?>

- `next()`: ជំរុញទ្រនិចអារេខាងក្នុងនៃអារេមួយ។

Ex: `<?php`
 // Sample array
 \$colors = array("red", "green", "blue", "orange", "yellow", "black");

 // Getting the values
 echo current(\$colors); // Prints: red
 echo next(\$colors); // Prints: green
 ?>

- `Pos ()`: ត្រឡប់ធាតុបច្ចុប្បន្ននៅក្នុងអារេមួយ។ ឈ្មោះក្លែងក្លាយនៃចរន្ត) មុខងារ។

Ex:

```
<?php
// Sample array
$colors = array("red", "green", "blue", "orange", "yellow", "black");

// Getting the current element
echo pos($colors); // Prints: red
?>
```

- `Prev ()`: ថយក្រោយទ្រនិចអារេខាងក្នុង។

Ex:

```
<?php
// Sample array
$colors = array("red", "green", "blue", "orange", "yellow", "black");

// Getting the values
echo current($colors); // Prints: red
echo next($colors);    // Prints: green
echo prev($colors);    // Prints: red
?>
```

- `range()`: បង្កើតអារេដែលមានជួរនៃធាតុ។

Ex:

```
<?php
// Creating range of numbers
$numbers = range(0, 5);
print_r($numbers);
?>
```

- `reset()`: កំណត់ទ្រនិចខាងក្នុងនៃអារេទៅធាតុទីមួយរបស់វា។

Ex:

```
<?php
// Sample array
$colors = array("red", "green", "blue", "orange", "yellow", "black");

// Getting the values
echo current($colors); // Prints: red
echo next($colors);    // Prints: green
echo reset($colors);   // Prints: red
?>
```

- `rsort()`: តម្រៀបអារេក្នុងលំដាប់បញ្ជាស ឬពីក្រោម។

Ex:

```
<?php
// Sample array
$fruits = array("apple", "orange", "mango", "banana", "kiwi");

// Sorting the fruits array alphabetically in descending order
rsort ($fruits);
print _r($fruits);
?>
```

- `shuffle()`:

- `sizeof()`: ត្រូវបានប្រើប្រាស់នៅក្នុងអារេមួយ។ ឈ្មោះក្លាយនៃមុខងារ `count()` ។

Ex:

```
<?php
// Sample array
$cars = array("Audi", "BMW", "Mercedes", "Volvo");

// Display array elements count
echo sizeof ($cars);
?>
```

- `sort()`: តម្រៀបអារេតាមលំដាប់ឡើង។

Ex:

```
<?php
// Sample array
$fruits = array("apple", "orange", "mango", "banana", "kiwi");

// Sorting the fruits array alphabetically in ascending order
sort($fruits);
print_r($fruits);
?>
```

- `Uasort ()`: តម្រៀបអារេដោយប្រើមុខងារប្រៀបធៀបដែលកំណត់ដោយអ្នកប្រើប្រាស់ និងរក្សាការភ្ជាប់លិខិតរៀបរៀង។

Ex:

```
<?php
// Define comparison function
function compare($a, $b){
    if($a == $b){
        return 0;
    }
    return ($a < $b) ? -1 : 1;
}

// Sample array
$numbers = array("a"=>2, "b"=>-1, "c"=>7, "d"=>-9, "e"=>5,
"f"=>-4);

// Sort numbers array using compare function
uasort($numbers, "compare");
print _r($numbers);
?>
```

- `Uksort ()`: តម្រៀបអារេដោយគ្រាប់ចុចដោយប្រើមុខងារប្រៀបធៀបដែលកំណត់ដោយអ្នកប្រើប្រាស់។

Ex:

```
<?php
// Define comparison function
function compare($a, $b){
    if($a == $b){
        return 0;
    }
    return ($a < $b) ? -1 : 1;
}

// Sample array
$numbers = array("a"=>1, "c"=>2, "f"=>3, "d"=>4, "b"=>5, "e"=>6);

// Sort numbers array using compare function
uksort($numbers, "compare");
print_r($numbers);
?>
```

- `Usort ()`: តម្រៀបអាទតាមតម្លៃដោយប្រើមុខងារប្រៀបធៀបដែលកំណត់ដោយអ្នកប្រើប្រាស់។

Ex:

```
<?php
// Define comparison function
function compare($a, $b){
    if($a == $b){
        return 0;
    }
    return ($a < $b) ? -1 : 1;
}

// Sample array
$numbers = array(2, 5, 4, 3, 6, 1);

// Sort numbers array using compare function
usort($numbers, "compare");
print_r($numbers);
?>
```

៤.១ PHP array_change_key_case () Function

- ❖ អនុគមន៍ array_change_key_case () ត្រូវបានប្រើដើម្បីផ្លាស់ប្តូរករណីនៃគ្រាប់ចុចទាំងអស់នៅក្នុងអារេទៅជាអក្សរតូច ឬអក្សរធំ។ Numbered ត្រូវបានទុកដូចនេះ។
 - តារាងខាងក្រោមសង្ខេបលម្អិតបច្ចេកទេសនៃមុខងារនេះ។
- ❑ Return Value ៖ ត្រឡប់អារេមួយដោយ lower key ឬ uppercased ឬ FALSE ប្រសិនបើអារេមិនមែនជាអារេ។
 - ❑ Version ៖ PHP 4.2+



៤.២.PHP array_chunk() Function បំបែកអារេមួយទៅជាកំណាត់។

❖array_chunk() បំបែកអារេមួយទៅជាកំណាត់។

តារាងខាងក្រោមសង្ខេបលម្អិតបច្ចេកទេសនៃមុខងារនេះ ។

❑Return Value៖ ត្រឡប់អារេដែលបានធ្វើលិបិក្រមជាលេខពហុវិមាត្រ ចាប់ផ្តើមដោយលេខសូន្យ ដោយវិមាត្រនីមួយៗមានធាតុទំហំ។

❑Version៖PHP 4.2+



៤.៣ PHP array _ combine() Function

- អនុគមន៍ array _ combine() បង្កើតអារេមួយដោយប្រើអារេមួយសម្រាប់ key និងមួយទៀតសម្រាប់ Value របស់វា។
 - តារាងខាងក្រោមសង្ខេបលម្អិតបច្ចេកទេសនៃមុខងារនេះ។
- ☐ Return Value៖ ត្រឡប់អារេមួយបញ្ចូលគ្នា FALSE ប្រសិនបើចំនួនធាតុសម្រាប់អារេនីមួយៗមិនស្មើគ្នា។
 - ☐ Changelog៖ កំណែមុន PHP 5.4.0 ចេញ E_WARNING ហើយត្រឡប់ FALSE សម្រាប់អារេទទេ។
 - ☐ Version៖ PHP 5+



៤. ៤ PHP array _ merge() Function

- បញ្ចូលអារេមួយ ឬច្រើនទៅក្នុងអារេមួយ។ function នេះរួមបញ្ចូលគ្នានូវធាតុនៃអារេមួយ ឬច្រើនជាមួយគ្នាតាមរបៀបដែល Value នៃមួយត្រូវបានបន្ថែមទៅចុងបញ្ចប់នៃមុន។ វាត្រឡប់ថ្មីមួយ។ អារេជាមួយធាតុរួមបញ្ចូលគ្នា។
- តារាងខាងក្រោមសង្ខេបលម្អិតបច្ចេកទេសនៃមុខងារនេះ។

❑ Return Value ៖ ត្រឡប់ អារេ ដែល បាន បញ្ចូល គ្នា

❑ Changelog ៖ ចាប់តាំងពី PHP 5.0 មុខងារនេះទទួលយកតែប៉ារ៉ាម៉ែត្រនៃប្រភេទអារេប៉ុណ្ណោះ។

❑ Version ៖ PHP 4+



៥. តំរៀប Array PHP Sorting Arrays

៥.១ PHP Functions For Sorting Arrays

- ❖ នៅក្នុងជំពូកមុន អ្នកបានសិក្សាពីសារៈសំខាន់នៃអារេរបស់ PHP ពេលគឺអារេជាអ្វី របៀបបង្កើត របៀបមើលរចនាសម្ព័ន្ធ របស់វា របៀបចូលប្រើធាតុរបស់វា ។ អ្នកអាចធ្វើសកម្មភាពជាច្រើនទៀតដូចជាការតម្រៀបអារេតាមអ្នកចង់។
- PHP ភ្ជាប់មកជាមួយនូវ built-in functions ដែលត្រូវបានរចនាឡើងជាពិសេសសម្រាប់ការតម្រៀបធាតុអារេក្នុងវិធីផ្សេងៗគ្នា ដូចជាតាមអក្ខរក្រម ឬជាលេខតាមលំដាប់ឡើង ឬ តាមលំដាប់ចុះ។ នៅទីនេះយើងនឹងស្វែងយល់ពីមុខងារទាំងនេះមួយ ចំនួនដែលប្រើជាទូទៅសម្រាប់ការតម្រៀបអារេ។
- sort() និង rsort () --សម្រាប់តម្រៀប indexed array
- Asort () និង arsort () - សម្រាប់តម្រៀបអារេពាក់ព័ន្ធតាមតម្លៃ
- ksort() និង krsort () - សម្រាប់តម្រៀបអារេពាក់ព័ន្ធតាមរយៈkey



```
<?php
// Define array
$colors = array("Red", "Green", "Blue", "Yellow");
// Sorting and printing array
sort($colors);
print_r($colors);
?>
```

- This print_r() statement gives the following output:

Array ([0] => Blue [1] => Green [2] => Red [3] => Yellow)

៥.២ Sorting Indexed Arrays in Ascending Order

- `sort()` function ត្រូវបានប្រើសម្រាប់តម្រៀបធាតុនៃ indexed អារេ តាមលំដាប់ឡើង (alphabetically សម្រាប់អក្សរ និង numerically សម្រាប់លេខ)។

```
?>
<?php
// Define array
$colors = array("Red", "Green", "Blue", "Yellow");
// Sorting and printing array
sort($colors);
print_r($colors);
?>
```

- This `print_r()` statement gives the following output:
- Array ([0] => Blue [1] => Green [2] => Red [3] => Yellow



៥.៣ Sorting Indexed Arrays in Descending Order

- `rsort()` function ត្រូវ បត្រូវបានប្រើសម្រាប់តម្រៀបធាតុនៃ index array តាមលំដាប់ចុះ (alphabetically សម្រាប់អក្សរ និង numerically សម្រាប់លេខ)។

```
<?php
// Define array
$numbers = array(1, 2, 2.5, 4, 7, 10);
// Sorting and printing array
rsort ($numbers);
print _r($numbers);
?>
```

This `print_r()` statement gives the following output:

```
Array ( [0] => 10 [1] => 7 [2] => 4 [3] => 2.5 [4] => 2 [5] => 1 )
```



៥.៤ Sorting Associative Arrays in Ascending Order By Value

- `asort()` function តម្រៀបធាតុនៃ associative array តាមលំដាប់ឡើងទៅតាមតម្លៃ។ វាដំណើរការដូចជា `sort()` ដែរ ប៉ុន្តែវាភ្ជាប់ទំនាក់ទំនងរវាង keys និងតម្លៃរបស់វានៅពេលតម្រៀប។

```
<?php
// Define array
$age = array("Peter"=>20, "Harry"=>14,
"John"=>45, "Clark"=>35);
// Sorting array by value and print
asort($age);
print_r($age);
?>
```

This `print _r()` statement gives the following output:

```
Array ( [Harry] => 14 [Peter] => 20 [Clark] => 35 [John] => 45 )
```



៥.៥ Sorting Associative Arrays in Descending Order By Value

- arsort () function តម្រៀបធាតុនៃassociativeតាមលំដាប់ចុះទៅតាមតម្លៃ។ វាដំណើរការដូច rsort () ប៉ុន្តែវា រក្សាទំនាក់ទំនងរវាង keys និងតម្លៃរបស់វា ខណៈពេលតម្រៀប។

```
<?php
// Define array
$age = array("Peter"=>20, "Harry"=>14,
"John"=>45, "Clark"=>35);
// Sorting array by value and print
arsort ($age);
print _r($age);
?>
```

This print_r() statement gives the following output:

Array ([John] => 45 [Clark] => 35 [Peter] => 20 [Harry] => 14)



៥.៦ Sorting Associative Arrays in Ascending Order By Key

- ksort() function តម្រៀបធាតុនៃ associative array តាមលំដាប់ឡើងតាមរយៈ key របស់វា។ វាក្រាស់ការផ្សារភ្ជាប់គ្នារវាង key និងតម្លៃរបស់វានៅពេលតម្រៀប ដូចគ្នានឹងមុខងារ asort()។

```
<?php
// Define array
$age = array("Peter"=>20, "Harry"=>14, "John"=>45,
"Clark"=>35);
// Sorting array by key and print
ksort($age);
print_r($age);
?>
```

This print_r() statement gives the following output:

Array ([Clark] => 35 [Harry] => 14 [John] => 45 [Peter] => 20)



៥.៧ Sorting Associative Arrays in Descending Order By Key

- krsort () functionតម្រៀបធាតុនៃ associative arrayតាមលំដាប់ចុះតាមរយៈkeyរបស់វា។ វារក្សាទំនាក់ទំនងរវាងkey និងតម្លៃរបស់វា ខណៈពេលដែលការតម្រៀប ដូចគ្នានឹង arsort () function។

```
<?php
// Define array
$age = array("Peter"=>20, "Harry"=>14, "John"=>45,
"Clark"=>35);
// Sorting array by key and print
krsort($age);
print_r($age);
?>
```

This print_r() statement gives the following output:

Array ([Peter] => 20 [John] => 45 [Harry] => 14 [Clark] => 35)

