

រៀបចំនិងកែសម្រួល​ពី ៖ ក្រុមទី១ ​ ​ ឆ្នាំ២០២៣

ព្រះរាជាណាចក្រកម្ពុជា

ជាតិ សាសនា ព្រះមហាក្សត្រ



ដេប៉ាតឺម៉ង៖ វិទ្យាសាស្រ្ដកុំព្យូទ័រ

សញ្ញាបត្រជាន់ខ្ពស់បច្ចេកទេស

Advance Internet Programming

វិទ្យាស្ថានពហុបច្ចេកទេសភូមិភាគតេជោសែនសៀមរាប

**Regional Polytechnic Institute Techo Sen Siem Reap**

១.តើសេរីទិន្នន័យ​ PHP ជាអ្វី

Array គឺជាអថេរស្មុគស្មាញដែលអនុញ្ញាតឲ្យយើងរក្សាទុកតម្លៃច្រើនជាងមួយឬក្រុមនៃតម្លៃក្រោមឈ្មោះអថេរតែមួយ។ការរក្សាទុកពណ៌ម្ដងក្នុងមួយអថេរអាចមើលខាងក្រោម៖

<?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**: indexedឬជាលេខរក្សាទុកធាតុអារេនីមួយៗជាមួយនឹងសន្ទស្សន៍ជាលេខ។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 key​ដែលត្រូវបានកំណត់ទៅតម្លៃអាចត្រូវបានកំណត់ដោយអ្នកប្រើ defined string។នៅក្នុងឧទាហរណ៍ ខាងក្រោមនេះ array ប្រើគន្លឹះជំនួសឲ្យលេខnumbers :

<?php

   // Define an associative array

$ages = array("Peter"=>22, "Clark"=>32, "John"=>28);

 ?>

**២.៣​ Multidimensional Arrays**

* **Multidimensional Arrays :** គឺជាប្រភេទនៃ Array ដែលរក្សាទុក Array មួយផ្សេងទៀតនៅindexនីមួយៗជំនួសឱ្យធាតុ។ ម្យ៉ាងវិញទៀត កំណត់អារេពហុវិមាត្រជាអារេនៃអារេ។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 មុខងារគឺជាផ្នែកនៃស្នូល PHP ដូច្នេះអ្នកអាចប្រើវានៅក្នុងស្គ្រីបរបស់អ្នកដោយគ្មាន ការដំឡើងបន្ថែម៖**

* **Array \_ change\_ key\_ case () : ផ្លាស់ប្ដូរករណីនៃ Key ទាំងអស់ក្នុង array ដើម្បីបន្ទាប caseឬពីលើ។**

<?php

  // Sample array

$persons = array("Harry"=>22, "Clark"=>32, "John"=>28);

 // Changing keys to lowercase

print \_ r(array \_ change \_key \_ case($persons));

?>

* **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។

Example:

<?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 key និង មួយ ទៀត សម្រាប់ Value របស់ វា ។

Example : <?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។

Example : <?php

// Sample array

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

// Counting array values

Print\_ r (array\_ count\_ values($letters));

?>

* Array \_ diff () :ប្រៀបធៀបតម្លៃអារេ ហើយ return difference។

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 () :ប្រៀបធៀបតម្លៃអារេដោយមានការត្រួតពិនិត្យall keyបន្ថែម និងreturn diference។

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 ():** ប្រៀបធៀប key arrays and return diference។

<?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​ \_ d​iff​​​​ \_​uassoc ​()**:ប្រៀបធៀបតម្លៃអារេ ជាមួយនឹងការពិនិត្យមើលkeyបន្ថែមដោយប្រើ function ប្រៀបធៀប គន្លឹះ ដែល កំណត់ ដោយ អ្នក ប្រើ ហើយ បង្ហាញ លទ្ធផល diference។

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 key array ដោយ ប្រើ key ដែល បាន កំណត់ ដោយ អ្នក ប្រើ function ប្រៀបធៀប and return diference។ 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 ()​:**បំពេញ array ជាមួយតម្លៃ បញ្ជាក់keys។

<?php

   // Defining keys array

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

   // Filling array

   $result = array \_f ill \_ keys($keys, "hello");

   print \_r($result);

?>

* **Array \_ filter () :** Theelements of array ដោយប្រើ 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 () :Return or change keyទាំងអស់ជាមួយនឹងValue ដែលពាក់ព័ន្ធរបស់វានៅក្នុងអារេមួយ។

Ex : <?php

  // Defining array

  $alphabets = array("a"=>"apple", "b"=>"ball", "c"=>"cat");

  // Flipping alphabets array

  $result = array \_f lip ($alphabets);

  print \_r($result);

?>

* Array \_intersect () :Comparation key array and return 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 () :Compares array value with additional key check , and return the matches .

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 ():Compare array key and return 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 () :Compare array value, with additional key check using a user\_ defined key comparison function, and return the matches.

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() :compare array key ,using a user \_ defined key comparison function and return the matches.

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() :ទទួលយកkeyដំបូងនៃអារេ។

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():ទទួលយកkeyចុងក្រោយនៃarray។

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 keyទាំងអស់ ឬសំណុំរងនៃkeyនៃ 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(): បញ្ចូនធាតុនៃarrayដែលបានផ្តល់ឱ្យទៅ 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(): បញ្ចូលarrayមួយឬច្រើនទៅក្នុង 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(): បញ្ចូលចំនួនធាតុដែលបានបញ្ជាក់ដោយitemដោយតម្លៃដែលបានបញ្ជាក់ទៅ 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 ហើយreturnតម្លៃនៃធាតុដែលបានដកចេញ។

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 (): Calculates the product of the value in an array.

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 keyមួយ ឬច្រើនពីarrayមួយ។

<?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 ទៅជាតម្លៃតែមួយដោយប្រើ function Callback ដែលកំណត់ដោយអ្នកប្រើប្រាស់។

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 arrayជាមួយធាតុក្នុងលំដាប់បញ្ច្រាស។

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 ធាតុមួយចេញពីarray។

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 ហើយជំនួសវាដោយអី្វផ្សេងទៀត។

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(): គណនាផលបូកនៃតម្លៃក្នុងarrayមួយ។

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 (): ប្រៀបធៀបតម្លៃអារេដោយប្រើ user defined ប្រៀបធៀប callback function ហើយreturn ភាពខុស

គ្នា។ 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 ():ប្រៀបធៀបតម្លៃអារេដោយប្រើ user defined function ប្រៀបធៀប callback function ជាមួយនឹងការពិនិត្យមើលkeyបន្ថែម និង returnភាពខុសគ្នា។

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 ():ប្រៀបធៀបkey និងតម្លៃនៃអារេដោយប្រើkeyដែលកំណត់ដោយអ្នកប្រើប្រាស់ពីរដាច់ដោយឡែក និងប្រៀបធៀបតម្លៃ callback function ហើយ returnភាពខុសគ្នា។

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 ():ប្រៀបធៀបតម្លៃអារេដោយប្រើ user defined function ប្រៀបធៀប callback functionហើយreturnការផ្គូផ្គង(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():ប្រៀបធៀបតម្លៃអារេដោយប្រើ user defined function ប្រៀបធៀប callback function ជាមួយនឹងការពិនិត្យkeyបន្ថែម និងreturn 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 ():ប្រៀបធៀបkey និងតម្លៃនៃអារេដោយប្រើ two separate user defined key និងប្រៀបធៀបតម្លៃ callback functionហើយនិង 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 ():យកតម្លៃស្ទួនចេញពីអារេមួយ។

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 ឬច្រើនទៅអារេដំបូង។

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 ():Returnតម្លៃទាំងអស់នៃអារេមួយ។

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 ():​តម្រៀបអារេដែលពាក់ព័ន្ធតាមតម្លៃក្នុងលំដាប់បញ្ច្រាសឬតាមលំដាប់ចុះ។

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

  asor t($alphabets);

  print\_ r($alphabets);

?>

* compact():បង្កើតarrayដែលមានអថេរ 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():ត្រឡប់ធាតុបច្ចុប្បន្ននៅក្នុងarrayមួយ ។

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 (): នាំចូលអថេរទៅក្នុងតារាងនិមិត្តសញ្ញាបច្ចុប្បន្នពីarrayមួយ។

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 (): ពិនិត្យមើលថាតើតម្លៃមាននៅក្នុងarray។

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(): ទាញយកkeyពី 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 keyក្នុង លំដាប់ បញ្ច្រាស 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(): កំណត់អថេរដូចជាប្រសិនបើវាជាអារេ។

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 (): តម្រៀបអារេដោយប្រើក្បួនដោះស្រាយ "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(): តម្រៀបអារេក្នុងលំដាប់បញ្ច្រាស 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);

?>

* Sizeof (): ត្រឡប់ចំនួនធាតុនៅក្នុងអារេមួយ។ឈ្មោះក្លែងក្លាយនៃមុខងារ 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 និងរក្សាការភ្ជាប់លិបិក្រម។

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);

?>

* 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 (): តម្រៀបអារេតាមតម្លៃដោយប្រើមុខងារប្រៀបធៀបដែលកំណត់ដោយអ្នកប្រើប្រាស់។

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

  usort($numbers, "compare");

  print\_r($numbers);

?>

**៤ .១ PHP array \_ change \_ key \_case () Function**

* អនុគមន៍ array \_ change key \_case () ត្រូវបានប្រើដើម្បីផ្លាស់ប្តូរករណីនៃ Keys ទាំងអស់នៅក្នុងអារេទៅជាអក្សរតូច ឬអក្សរធំ។numbered ត្រូវ បាន ទុក ដូច នេះ។
* តារាងខាងក្រោមសង្ខេបលម្អិតបច្ចេកទេសនៃមុខងារនេះ។
* **Return Value ៖** ត្រឡប់អារេមួយដោយ lower key ឬ uppercased ឬ FALSE ប្រសិនបើអារេមិនមែនជាអារេ។
* **Version៖**PHP 4.2+

**៤​ .២.PHP array \_chunk() Function**

array \_chunk() បំបែកអារេមួយទៅជាកំណាត់។

* តារាងខាងក្រោមសង្ខេបលម្អិតបច្ចេកទេសនៃមុខងារនេះ ។
* **Return Value៖** ត្រឡប់អារេដែលបានធ្វើលិបិក្រមជាលេខពហុវិមាត្រ ចាប់ផ្តើមដោយលេខសូន្យ ដោយវិមាត្រនីមួយៗមានធាតុទំហំ។
* **Version៖**PHP 4.2+

**៤.៣. PHP array \_ combine() Function**

* អនុគមន៍ array \_ combine() បង្កើតអារេមួយដោយប្រើអារេមួយសម្រាប់ key និងមួយទៀតសម្រាប់Value របស់វា។
* តារាងខាងក្រោមសង្ខេបលម្អិតបច្ចេកទេសនៃមុខងារនេះ។
* **Return Value៖** ត្រឡប់អារេរួមបញ្ចូលគ្នា 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។ functionនេះរួមបញ្ចូលគ្នានូវធាតុនៃអារេមួយ ឬច្រើនជាមួយគ្នាតាមរបៀបដែលValueនៃមួយត្រូវបានបន្ថែមទៅចុងបញ្ចប់នៃមុន។ វាត្រឡប់ថ្មីមួយ។ អារេជាមួយធាតុរួមបញ្ចូលគ្នា។
* តារាងខាងក្រោមសង្ខេបលម្អិតបច្ចេកទេសនៃមុខងារនេះ។
* Return Value៖ត្រឡប់ អារេ ដែល បាន បញ្ចូល គ្នា
* Changelog ៖ ចាប់តាំងពី PHP 5.0 មុខងារនេះទទួលយកតែប៉ារ៉ាម៉ែត្រនៃប្រភេទអារេប៉ុណ្ណោះ។
* Version៖PHP 4+

**៥.តំរៀបអារេPHP Sorting Array**

**៥.១​ .PHP Functions For Sorting Array**

* នៅក្នុងជំពូកមុន អ្នកបានសិក្សាពីសារៈសំខាន់នៃអារេរបស់ PHP ពោលគឺអារេជាអ្វី របៀបបង្កើត របៀបមើលរចនាសម្ព័ន្ធរបស់វា របៀបចូលប្រើធាតុរបស់វា ។ អ្នកអាចធ្វើសកម្មភាពជាច្រើនទៀតដូចជាការតម្រៀបអារេតាមអ្នកចង់។
* PHP ភ្ជាប់មកជាមួយនូវ built-in functionsដែលត្រូវបានរចនាឡើងជាពិសេសសម្រាប់ការតម្រៀបធាតុអារេក្នុងវិធីផ្សេងៗគ្នាដូចជាតាមអក្ខរក្រម ឬជាលេខតាមលំដាប់ឡើង ឬ តាមលំដាប់ចុះ។ នៅទីនេះយើងនឹងស្វែងយល់ពីមុខងារទាំងនេះមួយចំនួនដែលប្រើជាទូទៅសម្រាប់ការតម្រៀបអារេ។
* sort() និង rsort ​​​​​​​​​​() --សម្រាប់តម្រៀប indexed array
* Asort​ () និង arsort () - សម្រាប់តម្រៀបអារេពាក់ព័ន្ធតាមតម្លៃ
* ksort() និង krsort () - សម្រាប់តម្រៀបអារេពាក់ព័ន្ធតាមរយៈkey

**៥.២ Sorting Indexed Arrays in Ascending Order**

* sort() functionត្រូវបានប្រើសម្រាប់តម្រៀបធាតុនៃindexedអារេ តាមលំដាប់ឡើង (alphabeticallyសម្រាប់អក្សរ និង numericallyសម្រាប់លេខ)។

**៥.៣ Sorting Indexed Arrays in Descending Order**

* rsort() function ត្រូវ បត្រូវបានប្រើសម្រាប់តម្រៀបធាតុនៃ index arrayតាមលំដាប់ចុះ ( alphabeticallyសម្រាប់អក្សរ និង numericallyសម្រាប់លេខ)។

**៥.៤ Sorting Associative Arrays in Ascending Order By Value**

* asort() functionតម្រៀបធាតុនៃ associative arrayតាមលំដាប់ឡើងទៅតាមតម្លៃ។ វាដំណើរការដូចជា sort()ដែរ ប៉ុន្តែវារក្សាទំនាក់ទំនងរវាង keys និងតម្លៃរបស់វានៅពេលតម្រៀប។

**៥.៥ Sorting Associative Arrays in Descending Order By Value**

* arsort () function តម្រៀបធាតុនៃassociativeតាមលំដាប់ចុះទៅតាមតម្លៃ។ វាដំណើរការដូច rsort () ប៉ុន្តែវារក្សាទំនាក់ទំនងរវាង keys និងតម្លៃរបស់វា ខណៈពេលតម្រៀប។

**៥.៦ Sorting Associative Arrays in Ascending Order By Key**

* ksort() functionតម្រៀបធាតុនៃ associative arrayតាមលំដាប់ឡើងតាមរយៈ key របស់វា។ វារក្សាការផ្សារភ្ជាប់គ្នារវាង key និងតម្លៃរបស់វានៅពេលតម្រៀប ដូចគ្នានឹងមុខងារ asort()។

**៥.៧ Sorting Associative Arrays in Descending Order By Key**

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

ក្រុមទី១

១.ស៊ុន​ នៀវ

២.គត់ កាក់នីតា

៣.វ៉ង​ ​ស្រីវក

៤.ណាំង ស្រីនិច

៥.អួង វ៉ាន់ឌី