Assignment

1. $color = array(' blue ', 'green', ' black', ' white ', ' red');
   Write a script which will display the following string -
   "The memory of that scene for me is like a frame of film forever frozen at that moment: the red carpet, the green lawn, the white house, the leaden sky. The new president and his first lady. - Richard M. Nixon"
   and the words 'red', 'green' and 'white' will come from $color.
   Solution

```
<?php
$color== array(' blue ', 'green', ' black', ' white ', ' red');
echo "The memory of that scene for me is like a frame of film forever frozen at
that moment: the $color[4] carpet, the $color[1] lawn, the $color[3] house, the
leaden sky. The new president and his first lady. - Richard M. Nixon"."\n";
     ?>
```

2. $color = array('Black', 'green', 'red'')
Write a PHP script which will display the colors in the following way :
Output :
Black, green, red,

- green
- red
- black

Solution :

```
<?php
$color=array('white', 'green', 'red');
foreach ($color as $c)
{
echo "$c, ";
}
sort($color);
echo "<ul>";
foreach ($color as $y)
{
echo "<li>$y</li>";
}
echo "</ul>";
```

```
?>
```

3. $ceu = array("Italy"=>"Rome", "Luxembourg"=>"Luxembourg", "Belgium"=> "Brussels", "Denmark"=>"Copenhagen", "Finland"=>"Helsinki", "France" => "Paris", "Slovakia"=>"Bratislava", "Slovenia"=>"Ljubljana", "Germany" => "Berlin", "Greece" => "Athens", "Ireland"=>"Dublin", "Netherlands"=>"Amsterdam", "Portugal"=>"Lisbon", "Spain"=>"Madrid", "Sweden"=>"Stockholm", "United Kingdom"=>"London", "Cyprus"=>"Nicosia", "Lithuania"=>"Vilnius", "Czech Republic"=>"Prague", "Estonia"=>"Tallin", "Hungary"=>"Budapest", "Latvia"=>"Riga", "Malta"=>"Valetta", "Austria" => "Vienna", "Poland"=>"Warsaw") ;

Create a PHP script which displays the capital and country name from the above array $ceu. Sort the list by the capital of the country.

Sample Output :

```
The capital of Netherlands is Amsterdam
The capital of Greece is Athens
The capital of Germany is Berlin
-------------------------
-------------------------
```

Solution:

```php
<?php

$ceu=array( "Italy"=>"Rome", "Luxembourg"=>"Luxembourg",
"Belgium"=> "Brussels", "Denmark"=>"Copenhagen",
"Finland"=>"Helsinki", "France" => "Paris",
"Slovakia"=>"Bratislava", "Slovenia"=>"Ljubljana",
"Germany" => "Berlin", "Greece" => "Athens",
"Ireland"=>"Dublin", "Netherlands"=>"Amsterdam",
"Portugal"=>"Lisbon", "Spain"=>"Madrid",
"Sweden"=>"Stockholm", "United Kingdom"=>"London",
"Cyprus"=>"Nicosia", "Lithuania"=>"Vilnius",
"Czech Republic"=>"Prague", "Estonia"=>"Tallin",
"Hungary"=>"Budapest", "Latvia"=>"Riga","Malta"=>"Valetta",
"Austria" => "Vienna", "Poland"=>"Warsaw") ;
asort($ceu) ;
foreach($ceu as $country=>$capital)
{
echo "The capital of $country is $capital"."\n" ;
}

?>
```

4. $x = array(1, 2, 3, 4, 5);
Delete an element from the above PHP array. After deleting the element, integer keys must be normalized.
Sample Output :
array(5) { [0]=>int(1) [1]=>int(2) [2]=>int(3) [3]=>int(4) [4]=>int(5) }
array(4) { [0]=>int(1) [1]=>int(2) [2]=>int(3) [3]=>int(5) }

Solutions :

```
<?php
$x=array(1, 2, 3, 4, 5);
var_dump($x);
unset($x[3]);
$x=array_values($x);
echo '
';
var_dump($x);
?>
```

5. $color = array(4 => 'white', 6 => 'green', 11=> 'red');
Write a PHP script to get the first element of the above array.
Expected result : white

Solutions :

```
<?php
$color = array(4 => 'white', 6 => 'green', 11=> 'red');
echo reset($color)."\n";
?>
```

6. Write a PHP script that inserts a new item in an array in any position.
Expected Output :
Original array :
1 2 3 4 5
After inserting '$' the array is :
1 2 3 $ 4 5

Solution:

```
<?php
$original = array( '1','2','3','4','5' );
echo 'Original array : '."\n";
foreach ($original as $x)
```

```
{echo "$x ";}
$inserted = '$';
array_splice( $original, 3, 0, $inserted );
echo " \n After inserting '$' the array is : "."\n";
foreach ($original as $x)
{echo "$x ";}
echo "\n"
?>
```

7. Write a PHP script to sort the following associative array :
array("Sophia"=>"31","Jacob"=>"41","William"=>"39","Ramesh"=>"40") in
a) ascending order sort by value
b) ascending order sort by Key
c) descending order sorting by Value
d) descending order sorting by Key

Solution:

```
<?php
echo "
Associative array : Ascending order sort by value
";
$array2=array("Sophia"=>"31","Jacob"=>"41","William"=>"39","Ramesh"=>"40");
asort($array2);
foreach($array2 as $y=>$y_value)
{
echo "Age of ".$y." is : ".$y_value."
";
}
echo "
Associative array : Ascending order sort by Key
";
$array3=array("Sophia"=>"31","Jacob"=>"41","William"=>"39","Ramesh"=>"40");
ksort($array3);
foreach($array3 as $y=>$y_value)
{
echo "Age of ".$y." is : ".$y_value."
";
}
echo "
```

```
Associative array : Descending order sorting by Value
";
$age=array("Sophia"=>"31","Jacob"=>"41","William"=>"39","Ramesh"=>"40");
arsort($age);
foreach($age as $y=>$y_value)
{
echo "Age of ".$y." is : ".$y_value."
";
}
echo "
Associative array : Descending order sorting by Key
";
$array4=array("Sophia"=>"31","Jacob"=>"41","William"=>"39","Ramesh"=>"40");
krsort($array4);
foreach($array4 as $y=>$y_value)
{
echo "Age of ".$y." is : ".$y_value."
";
}
?>
```

8. Write a PHP script to calculate and display average temperature, five lowest and highest temperatures.

Recorded temperatures : 78, 60, 62, 68, 71, 68, 73, 85, 66, 64, 76, 63, 75, 76, 73, 68, 62, 73, 72, 65, 74, 62, 62, 65, 64, 68, 73, 75, 79, 73

Expected Output :

Average Temperature is : 70.6

List of seven lowest temperatures : 60, 62, 63, 63, 64,

List of seven highest temperatures : 76, 78, 79, 81, 85,

Solution

```
<?php
$month_temp = "78, 60, 62, 68, 71, 68, 73, 85, 66, 64, 76, 63, 81, 76, 73,
68, 72, 73, 75, 65, 74, 63, 67, 65, 64, 68, 73, 75, 79, 73";
$temp_array = explode(',', $month_temp);
$tot_temp = 0;
$temp_array_length = count($temp_array);
foreach($temp_array as $temp)
{
 $tot_temp += $temp;
}
 $avg_high_temp = $tot_temp/$temp_array_length;
 echo "Average Temperature is : ".$avg_high_temp."
";
sort($temp_array);
```

```
echo " List of five lowest temperatures :";
for ($i=0; $i< 5; $i++)
{
echo $temp_array[$i].", ";
}
echo "List of five highest temperatures :";
for ($i=($temp_array_length-5); $i< ($temp_array_length); $i++)
{
echo $temp_array[$i].", ";
}

?>
```

9. Write a PHP program to sort an array of positive integers using the Bead-Sort Algorithm. According to Wikipedia "Bead-sort is a natural sorting algorithm, developed by Joshua J. Arulanandham, Cristian S. Calude and Michael J. Dinneen in 2002. Both digital and analog hardware implementations of bead sort can achieve a sorting time of O(n); however, the implementation of this algorithm tends to be significantly slower in software and can only be used to sort lists of positive integers".

Input array : Array ([0] => 5 [1] => 3 [2] => 1 [3] => 3 [4] => 8 [5] => 7 [6] => 4 [7] => 1 [8] => 1 [9] => 3)
Expected Result : Array ([0] => 8 [1] => 7 [2] => 5 [3] => 4 [4] => 3 [5] => 3 [6] => 3 [7] => 1 [8] => 1 [9] => 1)

```
<?php
function columns($uarr)
{
$n=$uarr;
if (count($n) == 0)
 return array();
else if (count($n) == 1)
 return array_chunk($n[0], 1);
array_unshift($uarr, NULL);
 $transpose = call_user_func_array('array_map', $uarr);
return array_map('array_filter', $transpose);
}
function bead_sort($uarr)
{
foreach ($uarr as $e)
 $poles []= array_fill(0, $e, 1);
return array_map('count', columns(columns($poles)));
}
echo 'Original Array : '.'
';
print_r(array(5,3,1,3,8,7,4,1,1,3));
echo '
'.'After Bead sort : '.'
';
print_r(bead_sort(array(5,3,1,3,8,7,4,1,1,3)));
```

?>

10. Write a PHP program to merge (by index) the following two arrays.
Sample arrays :
$array1 = array(array(77, 87), array(23, 45));
$array2 = array("w3resource", "com");
Expected Output :

```
Array
(
[0] => Array
(
[0] => w3resource
[1] => 77
[2] => 87
)
[1] => Array
(
[0] => com
[1] => 23
[2] => 45
)
)
```

Solution:

```
<?php
$array1 = array(array(77, 87), array(23, 45));
$array2 = array("w3resource", "com");
function merge_arrays_by_index($x, $y)
{
$temp = array(); $temp[] = $x; if(is_scalar($y))
{
$temp[] = $y;
}
else
{
foreach($y as $k => $v)
{
$temp[] = $v;
}
}
return $temp;
}
echo '<pre>'; print_r(array_map('merge_arrays_by_index',$array2, $array1));
?>?>
```

11. Write a PHP function to change the following array's all values to upper or lower case.
Sample arrays :

$Color = array('A' => 'Blue', 'B' => 'Green', 'c' => 'Red');
Expected Output :
```
Values are in lower case.
Array ( [A] => blue [B] => green [c] => red )
Values are in upper case.
Array ( [A] => BLUE [B] => GREEN [c] => RED )
```

```php
<?php
function array_change_value_case($input, $ucase)
{
$case = $ucase;
$narray = array();
if (!is_array($input))
{
return $narray;
}
foreach ($input as $key => $value)
{
if (is_array($value))
{
$narray[$key] = array_change_value_case($value, $case);
 continue;
}
$narray[$key] = ($case == CASE_UPPER ? strtoupper($value) : strtolower($value));
}
return $narray;
}
$Color = array('A' => 'Blue', 'B' => 'Green', 'c' => 'Red');
echo 'Actual array ';
print_r($Color);
echo 'Values are in lower case.';
$myColor = array_change_value_case($Color,CASE_LOWER);
print_r($myColor);
echo 'Values are in upper case.';
$myColor = array_change_value_case($Color,CASE_UPPER);
print_r($myColor);
?>
```

12. Write a PHP script which displays all the numbers between 200 and 250 that are divisible by 4.
Note : Do not use any PHP control statement.
Expected Output : 200,204,208,212,216,220,224,228,232,236,240,244,248

```
<?php
 echo implode(",",range(200,250,4))."\n";
?>
```

13. Write a PHP script to get the shortest/longest string length from an array.
Sample arrays : ("abcd","abc","de","hjjj","g","wer")
Expected Output : The shortest array length is 1. The longest array length is 4.
solution:

```
<?php

$my_array = array("abcd","abc","de","hjjj","g","wer");
$new_array = array_map('strlen', $my_array);
// Show maximum and minimum string length using max() function and
min() function
echo "The shortest array length is " . min($new_array) .
". The longest array length is " . max($new_array).'.';
?>
```

14. Write a PHP script to generate unique random numbers within a range.
Sample Range : (11, 20)
Sample Output : 17 16 13 20 14 19 18 15 11 12

```
<?php
$n=range(11,20);
shuffle($n);
for ($x=0; $x< 10; $x++)
{
echo $n[$x].' ';
}
echo "\n"
?>
```

15. Write a PHP script to get the largest key in an array.
Solution

```
<?php
$ceu = array( "Italy"=>"Rome", "Luxembourg"=>"Luxembourg", "Belgium"=> "Brussels",
"Denmark"=>"Copenhagen", "Finland"=>"Helsinki", "France" => "Paris",
"Slovakia"=>"Bratislava",
"Slovenia"=>"Ljubljana", "Germany" => "Berlin", "Greece" => "Athens", "Ireland"=>"Dublin",
"Netherlands"=>"Amsterdam", "Portugal"=>"Lisbon", "Spain"=>"Madrid",
"Sweden"=>"Stockholm",
"United Kingdom"=>"London", "Cyprus"=>"Nicosia", "Lithuania"=>"Vilnius", "Czech
Republic"=>"Prague", "Estonia"=>"Tallin", "Hungary"=>"Budapest", "Latvia"=>"Riga",
"Malta"=> "Valetta","Austria" => "Vienna", "Poland"=>"Warsaw") ;
$max_key = max( array_keys( $ceu) );
echo $max_key."\n";
?>
```

16. Write a PHP function that returns the lowest integer that is not 0.

```
<?php
function min_values_not_zero(Array $values)
{
return min(array_diff(array_map('intval', $values), array(0)));
}
print_r(min_values_not_zero(array(-1,0,1,12,-100,1))."\n");
?>
```

17. Write a PHP function to floor decimal numbers with precision.
Note: Accept three parameters number, precision, and $separator
Sample Data :
1.155, 2, "."
100.25781, 4, "."
-2.9636, 3, "."

Expected Output :
1.15
100.2578
-2.964

Solution

```
<?php
function floorDec($number, $precision, $separator)
{
$number_part=explode($separator, $number);
$number_part[1]=substr_replace($number_part[1],$separator,$precision,0
);
if($number_part[0]>=0)
{$number_part[1]=floor($number_part[1]);}
else
{$number_part[1]=ceil($number_part[1]);}

$ceil_number= array($number_part[0],$number_part[1]);
return implode($separator,$ceil_number);
}
print_r(floorDec(1.155, 2, ".")."\n");
print_r(floorDec(100.25781, 4, ".")."\n");
print_r(floorDec(-2.9636, 3, ".")."\n");
?>
```

18. Write a PHP script to print "second" and Red from the following array.
Sample Data :
$color = array ("color" => array ("a" => "Red", "b" => "Green", "c" => "White"),
"numbers" => array (1, 2, 3, 4, 5, 6),
"holes" => array ("First", 5 => "Second", "Third"));

```
<?php
$color = array ( "color" => array ( "a" => "Red", "b" => "Green", "c"
=> "White"),
"numbers" => array ( 1, 2, 3, 4, 5, 6 ),
"holes" => array ( "First", 5 => "Second", "Third"));
echo $color["holes"][5]."\n"; // prints "second"
echo $color["color"]["a"]."\n"; // prints "Red"
?>
```

19. Write a PHP function to sort an array according to another array acting as a priority list.

```
<?php

function list_cmp($a, $b)
{
  global $order;

  foreach($order as $key => $value)
    {
      if($a==$value)
        {
          return 0;
          break;
        }

      if($b==$value)
        {
          return 1;
          break;
        }
    }
}

$order[0] = 1;
$order[1] = 3;
$order[2] = 4;
$order[3] = 2;

$array[0] = 2;
$array[1] = 1;
$array[2] = 3;
$array[3] = 4;
$array[4] = 2;
$array[5] = 1;
$array[6] = 2;

usort($array, "list_cmp");

print_r($array);
?>
```

20. Write a PHP script to sort the following array by the day (page_id) and username.

```
<?php

$arra[0]["transaction_id"] = "2025731470";
$arra[1]["transaction_id"] = "2025731450";
$arra[2]["transaction_id"] = "1025731456";
$arra[3]["transaction_id"] = "1025731460";
$arra[0]["user_name"] = "Sana";
$arra[1]["user_name"] = "Illiya";
$arra[2]["user_name"] = "Robin";
$arra[3]["user_name"] = "Samantha";

//convert timestamp to date
function convert_timestamp($timestamp){
    $limit=date("U");
    $limiting=$timestamp-$limit;
    return date ("Ymd", mktime (0,0,$limiting));
}
//comparison function
function cmp ($a, $b) {
    $l=convert_timestamp($a["transaction_id"]);
    $k=convert_timestamp($b["transaction_id"]);
    if($k==$l){
        return strcmp($a["user_name"], $b["user_name"]);
    }else{
        return strcmp($k, $l);
    }
}
//sort array
usort($arra, "cmp");

//print sorted info
while (list ($key, $value) = each ($arra)) {
    echo "\$arra[$key]: ";
    echo $value["transaction_id"];
    echo " user_name: ";
    echo $value["user_name"];
    echo "\n";
}
?>
```

21. Write a PHP program to sort a multi-dimensional array set by a specific key.

```
<?php
function column_Sort($unsorted, $column) {
    $sorted = $unsorted;
    for ($i=0; $i < sizeof($sorted)-1; $i++) {
      for ($j=0; $j<sizeof($sorted)-1-$i; $j++)
        if ($sorted[$j][$column] > $sorted[$j+1][$column]) {
          $tmp = $sorted[$j];
          $sorted[$j] = $sorted[$j+1];
          $sorted[$j+1] = $tmp;
      }
    }
    return $sorted;
}
$my_array = array();
$my_array[0]['name'] = 'Sana';
$my_array[0]['email'] = 'sana@example.com';
$my_array[0]['phone'] = '111-111-1234';
$my_array[0]['country'] = 'USA';

$my_array[1]['name'] = 'Robin';
$my_array[1]['email'] = 'robin@example.com';
```

```
$my_array[1]['phone'] = '222-222-1235';
$my_array[1]['country'] = 'UK';

$my_array[2]['name'] = 'Sofia';
$my_array[2]['email'] = 'sofia@example.com';
$my_array[2]['phone'] = '333-333-1236';
$my_array[2]['country'] = 'India';
print_r(column_Sort($my_array, 'name'));
?>
```

22. Write a PHP script to sort an array using case-insensitive natural ordering.

```
<?php
$colors = array(
    "color1", "color20", "color3", "color2"
);
sort($colors, SORT_NATURAL | SORT_FLAG_CASE);
foreach ($colors as $key => $val) {
    echo "Colors[" . $key . "] = " . $val . "\n";
}
?>
```

23. Write a PHP function to sort entity letters.

```
<?php
function entity_sort($my_array) {
  $total = count($my_array);
  for ($i=0;$i<$total;$i++) {
    if ($my_array[$i]{0} == '&') {
      $my_array[$i] = $my_array[$i]{1}.$my_array[$i];
    } else {
      $my_array[$i] = $my_array[$i]{0}.$my_array[$i];
    }
  }
  sort($my_array);
    for ($i=0;$i<$total;$i++) {
    $my_array[$i] = substr($my_array[$i],1);
  }
    return $my_array;
  }
$arr = array(" ","&", "<");
print_r(entity_sort($arr));
?>
```

24. Write a PHP function to shuffle an associative array, preserving key, value pairs.

```
<?php
function shuffle_assoc($my_array)
     {
        $keys = array_keys($my_array);

        shuffle($keys);

        foreach($keys as $key) {
            $new[$key] = $my_array[$key];
        }

        $my_array = $new;

        return $my_array;
    }

$colors = array("color1"=>"Red", "color2"=>"Green", "color3"=>"Yellow");

print_r(shuffle_assoc($colors));
?>
```

25. Write a PHP function to generate a random password (contains uppercase, lowercase, numeric and other) using shuffle() function.

```
<?php
function rand_Pass($upper = 1, $lower = 5, $numeric = 3, $other = 2) {

  $pass_order = Array();
  $passWord = '';

  //Create contents of the password
  for ($i = 0; $i < $upper; $i++) {
    $pass_order[] = chr(rand(65, 90));
  }
  for ($i = 0; $i < $lower; $i++) {
    $pass_order[] = chr(rand(97, 122));
  }
  for ($i = 0; $i < $numeric; $i++) {
    $pass_order[] = chr(rand(48, 57));
  }
  for ($i = 0; $i < $other; $i++) {
    $pass_order[] = chr(rand(33, 47));
  }

  //using shuffle() to shuffle the order
  shuffle($pass_order);

  //Final password string
  foreach ($pass_order as $char) {
     $passWord .= $char;
  }
  return $passWord;
}
echo "\n"."Generated Password : ".rand_Pass()."\n";
?>
```

26. Write a PHP script to sort an array in reverse order (highest to lowest).

```
<?php
$colors = array("Red", "Orange", "Black", "White");
rsort($colors);
print_r($colors);
?>
```

27. Write a PHP program to generate an array with a range taken from a string.

```
<?php

function string_range($str1)

{

  preg_match_all("/([0-9]{1,2})-?([0-9]{0,2}) ?,?;?/", $str1, $a);

  $x = array ();

  foreach ($a[1] as $k => $v)

  {

    $x  = array_merge ($x, range ($v,
(empty($a[2][$k])?$v:$a[2][$k])));

  }

  return ($x);

}

$test_string = '1-2 18-20 9-11';

print_r(string_range($test_string));

?>
```

28. Write a PHP program to get the index of the highest value in an associative array.

```
<?php
$x = array(
   'value1' => 3021,
   'value2' => 2365,
   'value3' => 5215,
   'value4' => 5214,
   'value5' => 2145);
reset($x);   // optional.
arsort($x);
$key_of_max = key($x);
echo "Index of the highest value : ".$key_of_max."\n";
?>
```

29. Write a PHP function to search a specified value within the values of an associative array.

```
<?php
function arraysearch($arra1, $search)
{
reset($arra1);
while (list ($key, $val) = each ($arra1))
{
if (preg_match ("/$search/i", $val))
{
  echo $search." has found in ".$key."\n";
}
else
{
   echo $search." has not found in ".$key."\n";
}
}
}
$exercises = array("part1"=>"PHP array", "part2"=>"PHP String", "part3"=>"PHP Math");
arraysearch($exercises, "Math");
?>
```