

# ԳՈՐԾՈՂՈՒԹՅՈՒՆՆԵՐ ԶԱՆԳՎԱԾՆԵՐԻ ՀԵՏ

## Մաս 1

### Is\_array () ֆունկցիան

is\_array () ֆունկցիան վերադարձնում է true, եթե փոփոխականը հանդիսանում է զանգված, և false՝ հակառակ դեպքում:

```
$isar = is_array($technics);  
echo ($isar==true)?"զանգված է":"զանգված չէ";
```

### Count/sizeof ֆունկցիան

count/sizeof ֆունկցիան վերադարձնում է զանգվածի տարրերի քանակը:

```
$number = count($technics);  
// նույնն է, ինչ որ  
// $number = sizeof($technics);  
echo "technics զանգվածում կա $number տարր";
```

### Shuffle ֆունկցիան

Shuffle ֆունկցիան խառնում է զանգվածի տարրերը պատահականորեն:

```
$os=array("Windows 95", "Windows XP", "Windows Vista", "Windows 7", "Windows 8",  
"Windows 10");  
shuffle($os);  
print_r($os);  
// արտածված տարրերակներից մեկը.  
// Array ( [0]=>Windows 7 [1]=>Windows 10 [2]=>Windows XP [3]=>Windows 8  
// [4]=>Windows Vista [5] => Windows 95 )
```

### Compact ֆունկցիան

compact ֆունկցիան թույլ է տալիս փոփոխականների հավաքածուից ստեղծել ասոցիացված զանգված, որտեղ բանալիներ կհանդիսանան փոփոխականների անունները:

```
<?php
```

```

$model = "Apple II";
$producer = "Apple";
$year = 1978;

$data = compact('model', 'producer', 'year');
print_r($data);
// կարտածվի
// Array ( [model] => Apple II [producer] => Apple [year] => 1978 )
?>

```

## ՉԱՆԳՎԱԾՆԵՐԻ ԴԱՍԱՎՈՐՈՒՄ ԸՍՏ ԱՐԺԵՔՆԵՐԻ

PHP- ում կա երկու տիպի դասավորում՝ այբբենական կարգով տողերի դասավորում և աճման / նվազման կարգով թվերի դասավորում:

Եթե կարգավորվող արժեքները տող են, ապա դրանք դասավորում են այբբենական կարգով, եթե թվեր են, ապա դրանք դասավորվում են աճման կարգով:

PHP- ն տեսակավորման տիպը ընտրում է ինքնաբերաբար:

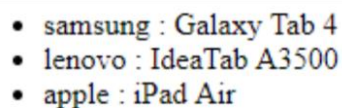
Աճման կարգով դասավորելու համար կիրառվում է `asort` ֆունկցիան:

```

<?php
$tablets = array("lenovo" => "IdeaTab A3500",
                 "samsung" => "Galaxy Tab 4",
                 "apple" => "iPad Air");
asort($tablets);

echo "<ul>";
foreach ($tablets as $key => $value)
{
    echo "<li>$key : $value</li>";
}
echo "</ul>";
?>

```



```

• samsung : Galaxy Tab 4
• lenovo : IdeaTab A3500
• apple : iPad Air

```

Լրացուցիչ պարամետրի օգնությամբ կարելի է հստակորեն նշել դասավորման տիպը: Այս պարամետրը կարող է ընդունել երեք արժեք.

`SORT_REGULAR` - տեսակավորման տիպը ընտրում է ինքնաբերաբար:

SORT\_NUMERIC - թվերի դասավորում:

```
asort($tablets, SORT_STRING);
```

SORT\_STRING - տողերի դասավորում:

Զանգվածը հակառակ հերթականությամբ դասավորելու համար օգտագործվում է arsort ֆունկցիան:

## ԶԱՆԳՎԱԾՆԵՐԻ ԴԱՍԱՎՈՐՈՒՄ ԸՍՏ ԲԱՆԱԼԻԻ

Asort ֆունկցիան կատարում է դասավորում ըստ տարրերի արժեքների, բայց գոյություն ունի նաև տեսակավորում ըստ բանալիների: Այն ներկայացված է ksort ֆունկցիայով:

```
ksort($tablets, SORT_STRING);
```

Բանալիների դասավորում հակառակ հերթականությամբ կատարվում է krsort ֆունկցիայով:

```
krsort($tablets);
```

## ԶԱՆԳՎԱԾՆԵՐԻ ԲՆԱԿԱՆ ԴԱՍԱՎՈՐՈՒՄ

Չնայած նրան, որ վերը նկարագրված դասավորման գործառույթները կատարելապես կատարում են իրենց աշխատանքը, բայց նրանց հնարավորությունները դեռ բավարար չեն: Օրինակ, դասավորեք աճման կարգով հաջորդ զանգված:

```
<?php
```

```
$os = array("Windows 7", "Windows 8", "Windows 10");
asort($os);
print_r($os);
// Array ( [2] => Windows 10 [0] => Windows 7 [1] => Windows 8 )
```

```
?>
```

Քանի որ արժեքները ներկայացնում են տողեր, PHP-ն դրանք դասավորում է այբբենական կարգով: Սակայն նման տեսակավորումը հաշվի չի առնում թվերը և ռեզիստրը: Հետևաբար, "Windows 10" արժեքը կգնա հենց սկզբից, այլ ոչ թե վերջում:

Այս խնդիրը լուծելու համար PHP-ում կա natasort() ֆունկցիան, որը կատարում է բնական դասավորում:

```
<?php
    $os = array("Windows 7", "Windows 8", "Windows 10");
    natsort($os);
    print_r($os);
    // Array ( [0] => Windows 7 [1] => Windows 8 [2] => Windows 10)
?>
```

Եթե մեզ պետք է, որ տեսակավորումը հաշվի չառնի ռեգիստրը, ապա մենք կարող ենք կիրառել natcasesort() ֆունկցիան՝ natcasesort(\$os):

## ԻՆՉՊԵՍ ԱՎԵԼԱՑՆԵԼ ՏԱՐԴԸ ԱՍՈՑԻԱՏԻՎ ԶԱՆԳՎԱԾԻ ՄԿԶԲՈՒՄ

Օրինակ, տրված է զանգված.

```
$arr = ['key1' => 'value1', 'key2' => 'value2'];
```

Եթե կատարենք \$arr['key0'] = 'value0' գործողությունը կստանանք.

Array

```
(
    [key1] => value1
    [key2] => value2
    [key0] => value0
)
```

Որպեսզի տարրը հայտնվի ճիշտ տեղում, պետք է օգտագործել կցման օպերատորը.

```
$arr1 = ['key0' => 'value0'] + $arr1;
```

## Array\_combine ֆունկցիան

array\_combine(), ֆունկցիան երկու գոյություն ունեցող զանգվածներից ստեղծում է նոր զանգված: Առաջին զանգվածը օգտագործում է բանալիներ ստեղծելու համար, երկրորդը՝ արժեքներ:

```
$keys = ['sky', 'grass', 'orange'];
$values = ['blue', 'green', 'orange'];
```

```
$array = array_combine($keys, $values);  
print_r($array);
```

```
// Array  
// (  
//   [sky] => blue  
//   [grass] => green  
//   [orange] => orange  
// )
```

### **Array\_values, array\_keys և array\_flip ֆունկցիաները**

array\_values() ֆունկցիան ասոցիատիվ զանգվածից դուրս է բերում արժեքները, array\_keys()-ը վերադարձնում է միայն զանգվածի բանալիները, իսկ array\_flip()-ը փոխում է տեղերով բանալիները և արժեքները:

```
print_r(array_keys($array)); // ['sky', 'grass', 'orange']  
print_r(array_values($array)); // ['blue', 'green', 'orange']  
print_r(array_flip($array));
```

```
// Array  
// (  
//   [blue] => sky  
//   [green] => grass  
//   [orange] => orange  
// )
```

### **Array\_unique ֆունկցիան**

Որպեսզի ստանանք զանգված միայն եզակի արժեքներով, պետք է կիրառել array\_unique() ֆունկցիան: Հարկ է նշել, որ ստացվող զանգվածում կներառվեն միայն հայտնաբերված առաջին տարրերը:

```
$array = [1, 1, 1, 1, 2, 2, 2, 3, 4, 5, 5];  
$uniques = array_unique($array);  
print_r($uniques);
```

```
// Array  
// (  
//   [0] => 1  
//   [4] => 2  
//   [7] => 3
```

```
// [8] => 4  
// [9] => 5  
// )
```

### **Array\_column () ֆունկցիան**

Array\_column () ֆունկցիան օգտակար կլինի, եթե անհրաժեշտ է բազմաչափ զանգվածի որոշակի սյունակ արտածել: Դա կարող է լինել SQL հարցման կամ CSV ֆայլի տվյալներ:

Դրա համար պետք է նշել զանգվածի և սյունակի անվանումը:

```
$array = [  
    ['id' => 1, 'title' => 'tree'],  
    ['id' => 2, 'title' => 'sun'],  
    ['id' => 3, 'title' => 'cloud'],  
];  
  
$ids = array_column($array, 'id');  
print_r($ids); // [1, 2, 3]
```