# CAP. 5
# VECTORI (ARRAYS) ÎN PHP

Vectorii (arrays) sunt un concept important în PHP. Un array reprezintă o structură de date ce poate stoca mai multe valori în același timp într-o singură "variabilă" (sub formă de listă ordonată). Aceștia pot fi folosiți pentru a stoca, de exemplu, o listă de numere sau o listă de șiruri de caractere (text). De asemenea, putem folosi un vector pentru a grupa zilele săptămânii într-o listă sau pentru a stoca toți utilizatorii aplicației noastre.

Mai jos aveți prezentată diferența dintre utilizarea unui vector și a unor variabile individuale pentru stocarea informațiilor.

```
// Inițializarea și utilizarea unui vector
$masini = array("Volvo", "BMW", "Toyota");
echo "Masini: " . $masini[0] . ", " . $masini[1] . " și " .
$masini[2] . ".";

// Echivalent prin utilizarea mai multor variabile
$masina1 = "Volvo";
$masina2 = "BMW";
$masina3 = "Toyota";
```

Ați putea spune că nu este o diferență foarte mare între cele două variante de stocare a informațiilor, dar daca extrapolați exemplul la mult mai multe date (ex: 100, 1000, 1000.000) lucrurile se complică. Vectorii mai prezintă, de asemenea, un avantaj privind parcurgerea informațiilor conținute: pot fi utilizați foarte facil în cadrul structurilor repetitive.

Dimensiunea unui vector poate fi aflată foarte ușor cu ajutorul funcției count(). Cu ajutorul acestei informații putem construi structuri repetitive ce ne permit să iterăm peste valorile conținute:

```
$masini = array("Volvo", "BMW", "Toyota");
echo "Nr. masini: " . count($masini) . "<br>";

for ($i = 0; $i < count($masini); $i++) {
    echo $masini[$i] . "<br>";
}
```

În cadrul acestui curs veți învăța cum să definiți structuri de date de tip array, cum să lucrați cu aceștia și ce funcții importante și utile există.

## 5.1 Tipuri de arrays

În PHP, există trei tipuri de vectori (arrays):

- **Indexed arrays** (vectori cu indecși numerici);
- **Associative arrays** (vectori asociativi);
- **Multidimensional arrays** (vectori multidimensionali).

### 5.1.1 Indexed array

Acest tip de vectori poate fi utilizat pentru a stoca orice tip de valori, dar indecși sunt întotdeauna numere. În mod implicit, primul indice începe de la zero. Acești vectori pot fi creați în moduri diferite, așa cum se observa în următoarele exemple:

```
// Primul mod de a crea un vector
$vector_unu = array("Ion", "Anton", "Radu", "Vasilica", "Gheorghe",
"Cristian");

// Al doilea mod de a crea un vector
$vector_doi[0] = "George";
$vector_doi[1] = "Ion";
$vector_doi[2] = "Maria";
$vector_doi[3] = "Ioana";

// Al treilea mod de a crea un vector
$vector_trei = ["Marius", "Mirel", "Mihaela", "Matei"];

// Accesarea elementelor
echo $vector_unu[2] . "<br>";
echo $vector_doi[1] . "<br>";
echo $vector_trei[0] . "<br>";
```

Structura repetitivă **foreach** este destinată parcurgerii vectorilor deoarece cheile vectorilor în PHP nu sunt neapărat valori numerice sau dacă sunt nu sunt neapărat valori consecutive, deci în anumite cazuri este imposibil să folosim o structură **for** simplă.

```
$vector_unu = array("Ion", "Anton", "Radu", "Vasilica", "Gheorghe",
"Cristian");

foreach ($vector_unu as $val) {
    echo $val . "<br>";
}
```

### 5.1.2 Array-uri asociative

Vectorii asociativi sunt similari cu cei indexați, dar în loc de indecși numerici, fiecare valoare poate fi asociată cu o cheie de tip **string**, definită de utilizator.

```
// Primul mod de a crea un vector
$vector_unu = array(
    "nume1" => "Ion",
    "nume2" => "Anton",
    "nume3" =>"Radu",
    "nume4" =>"Vasilica"
);
```

```php
// Al doilea mod de a crea un vector
$vector_doi["var"] = "George";
$vector_doi["unchi"] = "Ion";
$vector_doi["vara"] = "Maria";
$vector_doi["matusa"] = "Ioana";

// Al treilea mod de a crea un vector
$vector_trei = [
    "coleg" =>"Marius",
    "amic" =>"Mirel",
    "colega" =>"Mihaela",
    "vecin" =>"Matei"
];

// Accesarea elementelor
echo $vector_unu["nume2"] . "<br>";
echo $vector_doi["unchi"] . "<br>";
echo $vector_trei["amic"] . "<br>";
```

Parcurgerea unui vector asociativ se realizează foarte facil folosind structura **foreach**:

```php
$vector_unu = array("nume1" => "Ion", "nume2" => "Anton",
"nume3" =>"Radu", "nume4" =>"Vasilica");

foreach ($vector_unu as $cheie => $val) {
    echo "Cheie=" . $cheie . ", Valoare=" . $val;
}
```

### 5.1.3 Array-uri multidimensionale

Vectorii multidimensionali sunt structuri de date de tip array care stochează un alt array la fiecare index în loc de un singură variabilă (valoare). Cu alte cuvinte, putem defini vectorii multidimensionali ca fiind vectori de vectori (sau matrice).

PHP acceptă matrice multidimensionale care au două, trei, patru, cinci sau mai multe niveluri. Cu toate acestea, matricele mai mari de trei niveluri sunt greu de gestionat pentru majoritatea oamenilor.

Dimensiunea unei matrice indică numărul de indici de care aveți nevoie pentru a selecta un element.

- Pentru o matrice bidimensională aveți nevoie de doi indici pentru a selecta un element;
- Pentru o matrice tridimensională aveți nevoie de trei indici pentru a selecta un element.

```php
// Mod de a crea un vector multidimensional
$masini = array (
    array("Volvo",22,18),
    array("BMW",15,13),
    array("Saab",5,2),
    array("Land Rover",17,15)
```

```
);

// Mod de a accesa valorile unui vector multidimensional
echo $masini[0][0].": In stock: ".$masini[0][1].", vândute:
".$masini[0][2].".<br>";
echo $masini[1][0].": In stock: ".$masini[1][1].", vândute:
".$masini[1][2].".<br>";
echo $masini[2][0].": In stock: ".$masini[2][1].", vândute:
".$masini[2][2].".<br>";
echo $masini[3][0].": In stock: ".$masini[3][1].", vândute:
".$masini[3][2].".<br>";

// Mod de iterare prin valorile unui vector multidimensional
for ($row = 0; $row < 4; $row++) {
    echo "<p><b>Numărul rândului $row</b></p>";
    echo "<ul>";
    for ($col = 0; $col < 3; $col++) {
        echo "<li>".$masini [$row][$col]."</li>";
    }
    echo "</ul>";
}
```

**Observație:** Se pot folosi și celelalte moduri pentru definirea unui vector multidimensional (prezentate pentru vectorii indexați sau asociativi).

## 5.2 Sortarea vectorilor

Principalele funcții de sortare a unui vector în PHP sunt :

- **sort()** - sortează vectorii în ordine crescătoare;
- **rsort()** - sortează vectorii în ordine descrescătoare;
- **asort()** - sortează vectorii asociativi în ordine crescătoare, în funcție de valoare;
- **ksort()** - sortează vectorii asociativi în ordine crescătoare, în funcție de cheie;
- **arsort()** - sortează vectorii asociativi în ordine descrescătoare, în funcție de valoare;
- **krsort()** - sortează vectorii asociativi în ordine descrescătoare, în funcție de cheie.

### Funcția sort()

Funcția sort() reorganizează valorile stocate într-un vector în ordine crescătoare.

```
$masini = array("Volvo", "BMW", "Toyota");
sort($masini); // Sortarea modifică vectorul primit ca parametru

// Afișarea vectorului
$clength = count($masini);
for($x = 0; $x < $clength; $x++) {
  echo $masini[$x];
  echo "<br>";
}
```

### Funcția rsort()

Funcția rsort() reorganizează valorile stocate într-un vector în ordine descrescătoare.

```
$masini = array("Volvo", "BMW", "Toyota");
rsort($masini); // Sortarea modifică vectorul primit ca parametru
```

### Funcția asort()

Funcția asort() sortează vectorii asociativi în ordine crescătoare, în funcție de valoare.

```
$vârstă = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
asort($vârstă);
// Afișarea vectorului asociativ
foreach($vârstă as $x => $x_value) {
  echo "Cheie=" . $x . ", Valoare=" . $x_value . "<br>";
}
```

### Funcția ksort()

Funcția ksort() sortează vectorii asociativi în ordine crescătoare, în funcție de cheie.

```
$vârstă = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
ksort($vârstă);
```

### Funcția arsort()

Funcția arsort() sortează vectorii asociativi în ordine descrescătoare, în funcție de valoare.

```
$vârstă = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
arsort($vârstă);
```

### Funcția krsort()

Funcția krsort() sortează vectorii asociativi în ordine descrescătoare, în funcție de cheie.

```
$vârstă = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
krsort($vârstă);
```

## 5.3 Alte funcții predefinite pentru vectori

Funcțiile predefinite pentru vectori fac parte din nucleul PHP. Nu este nevoie de nici-o operație pentru a utiliza aceste funcții. Ele vă permit să accesați și să manipulați structurile de tip vector. Sunt acceptați atât vectorii simpli cât și multidimensionali. O listă cu cele mai utilizate funcții destinate manipulării vectorilor este prezentată în tabelul următor.

| Funcție | Descriere |
|---|---|
| array_chunk() | Împarte un vector mai mare în vectori mai mici cu dimensiunea specificată ca parametru |
| array_column() | Returnează valorile de pe o coloană specificată dintr-un vector multidimensional |
| array_combine() | Creează un vector asociativ folosind drept chei elementele dintr-un vector și drept valori elementele dintr-un alt vector |

| | |
|---|---|
| array_count_values() | Afișează frecvența elementelor unui vector |
| array_fill() | Adaugă o valoare dată pe toate pozițiile specificate ca parametru |
| array_intersect() | Compară doi vectori și returnează valorile comune |
| array_key_exists() | Verifică dacă cheia primită ca parametru există în vector |
| array_keys() | Returnează toate cheile unui vector |
| array_merge() | Combină unul sau mai mulți vectori în unul singur |
| array_pop() | Șterge ultimul element dintr-un vector |
| array_push() | Adaugă unul sau mai multe elemente la finalul unui vector |
| array_rand() | Returnează una sau mai multe chei la întâmplare dintr-un vector |
| array_search() | Caută o valoare într-un vector |
| array_slice() | Returnează o parte dintr-un vector |
| array_sum() | Returnează suma tuturor valorilor dintr-un vector |
| array_unique() | Șterge valorile duplicat dintr-un vector |
| array_values() | Returnează toate valorile dintr-un vector |
| in_array() | Verifică dacă o valoare se află într-un vector |

## Teme

1. Creați o funcție care primește 2 vectori ca parametri și returnează vectorul format din elementele comune ale celor 2 vectori.
2. Creați o funcție care primește ca parametru un vector și afișează câte dintre elementele sale conțin: 1 caracter, 2 caractere, 3 caractere, 4 caractere, 5 sau mai multe caractere.
3. Creați o funcție care primește 2 parametri. Primul parametru este un vector de vectori, iar al doilea parametru este un vector. Elementele parametrului 2 vor fi adăugate la finalul vectorilor conținuți de primul parametru dacă numărul de elemente inițiale este mai mic sau egal cu 3.
4. Creați o funcție care primește 1 parametru ce reprezintă un vector de vectori. Funcția va returna un vector ordonat descrescător format din toate elementele unice existente în vectorii parametrului.
5. Creați o funcție ce primește 3 parametri. Primul parametru este un vector asociativ existent, al doilea parametru va fi folosit pe post de cheie pentru al treilea parametru. Asocierea creata între parametrul 2 și 3 va fi adăugată în vectorul reprezentat de parametrul 1. Funcția va returna vectorul nou creat.
6. Creați o funcție care inițializează un vector indexat cu 10 elemente, la fiecare index se va stoca un vector asociativ cu următoarele chei: nume, oraș, telefon. Valoarea asociată fiecărei chei va fi "Null". Creați o altă funcție care primește 5 parametri (vector, index, nume, oraș, telefon). Aceasta va verifica dacă vectorul primit ca argument conține indexul primit și va popula vectorul asociativ aflat la indexul primit cu valorile pentru nume, oraș și telefon.
7. Creați o funcție care face o statistică a valorilor stocate la cheia oraș în vectorul definit anterior. Funcția va returna un vector asociativ ce folosește pe post de chei valorile de oraș găsite în vectorul primit ca argument, iar valorile asociate fiecărei chei reprezentă numărul de apariții în vectorul primit ca argument.