Pagini WEB cu PHP 4

Mihai Scorțaru, Claudiu Soroiu

În cadrul acestui articol dedicat limbajului de interpretare PHP vom începe prezentarea funcțiilor puse la dispoziție de acesta. Vom descrie funcțiile care pot fi utlizate pentru manipularea tablourilor.

Tablouri

PHP oferă numeroase funcții predefinite care ușurează prelucrarea tablourilor. Vom prezenta în continuare sintaxele acestor funcții și vom exemplifica modul în care acestea pot fi utilizate.

Funcția array_change_key_case

Această funcție realizează conversia tuturor literelor care apar în cadrul denumirilor cheilor elementelor unui tablou.

Funcția are doi parametri: un tablou asupra căruia se vor efectua modificările și un întreg care arată dacă se vor converti toate literele mari în litere mici sau invers.

Pentru acest parametru pot fi utilizate constantele predefinite `CASE_UPPER` (conversie la litere mari) și `CASE_LOWER` (conversie la litere mici).

Acest parametru poate lipsi, caz în care va avea valoarea implicită `CASE_LOWER`, deci va realiza conversia tuturor literelor mari în litere mici.

Următorul document *PHP* ilustrează modul în care poate fi utilizată această funcție. Efectul este prezentat în figura 1.

```
<HTML>
  <HEAD>
    <TITLE>
      Funcția array_change_key_case
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      $a = array("Gazeta" => "Gazeta" , "de"
                    => "de" , "Informatică" =>
                              "Informatică");
      echo "<TABLE cellpadding = 5>";
      echo "<TR><TD><B>Tabloul inițial:"
      echo "</B><TD><TT>";
      print_r($a);
      echo "</TT><TR><TD><B>Tabloul după "
      echo "conversia implicită:"
      echo "</B><TD><TT>";
      $b = array_change_key_case($a);
      print_r($b);
      echo "</TT><TR><TD><B>Tabloul după "
      echo "conversia la litere mari:"
      echo "</B><TD><TT>";
      $b = array_change_key_case($a,CASE_UPPER);
      print_r($b);
      echo "</TT><TR><TD><B>Tabloul după "
      echo "conversia la litere mici:"
      echo "</B><TD><TT>";
      $b = array_change_key_case($a,CASE_LOWER);
      print_r($b);
      echo "</TT></TABLE>"
    ?>
  </BODY>
</HTML>
```

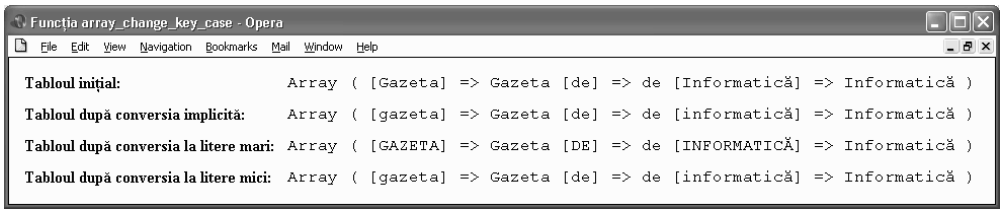

Figura 1: Funcția array_change_key_case

După cum se poate vedea în figură, funcția are efect doar asupra cheilor elementelor, valorile acestora rămânând nemodificate.

Funcția array_chunks

Această funcție poate fi utilizată pentru a împărți un tablou în mai multe bucăți (*chunks*). Ea are trei parametri, și anume: vectorul care urmează a fi împărțit, dimensiunea unei bucăți (numărul de elemente) și un indicator care arată dacă se păstrează sau nu cheile elementelor (acesta poate lipsi; în acest caz are valoarea implicită **false**, deci nu se vor păstra cheile elementelor).

În cazul în care numărul elementelor tabloului inițial nu este multiplu al dimensiunii unei bucăți, atunci ultima bucată va fi mai "scurtă". Funcția va returna un tablou al bucăților (un tablou de tablouri).

Prezentăm în continuare un document *PHP* care ilustrează modul în care poate fi utilizată această funcție. Efectul este prezentat în figura 2.

```
<HTML>
  <HEAD>
    <TITLE>
      Funcția array_chunks
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      $vector = array(
       "a","b","c","d","e");
      echo "<PRE>";
      print_r(array_chunk
                ($vector,2));
      echo "<P>";
      print_r(array_chunk(
           $vector,2,TRUE));
      echo "</PRE>";
    ?>
  </BODY>
</HTML>
```

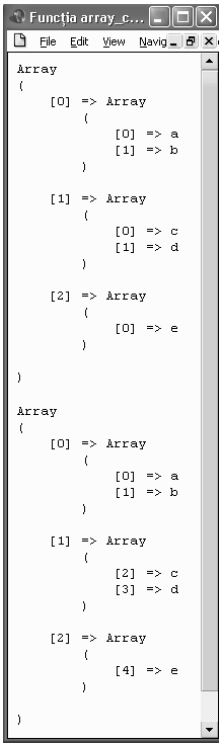

Figura 2: Funcția array_chunks

Funcția array_count_values

Această funcție are ca parametru un tablou și returnează un alt tablou ale cărui elemente au chei date de valorile elementelor tabloului transmis ca parametru. Pentru fiecare valoare se va genera un singur element (chiar dacă valoarea apare de mai multe ori), iar valoarea elementului respectiv este dată de numărul aparițiilor.

De exemplu, în urma execuției unei secvențe de forma:

```
$tablou = array(1,"hello",1,"world","hello");
$rez = array_count_values ($tablou));
```

vectorul `$rez` va conține trei elemente, acestea vor avea cheile `1`, `hello` și `world`, iar valorile vor fi `2`, `2`, respectiv `1`.

Funcțiile array_diff și array_diff_assoc

Uneori dorim să eliminăm dintr-un tablou anumite elemente; în acest scop pot fi utilizate funcțiile `array_diff()` și `array_dif_assoc()`.

Acestea au ca parametri mai multe tablouri și returnează un tablou care conține toate elementele din tabloul reprezentat de primul parametru care nu apar în nici unul dintre tablourile corespunzătoare celui de-al doilea parametru.

În cazul funcției `array_diff()` sunt eliminate elemente din primul tablou dacă în celelalte există elemente cu aceeași valoare.

În cazul funcției `array_diff_assoc()` sunt eliminate elemente din primul tablou dacă în celelalte există elemente cu aceeași valoare și aceeași cheie.

Funcția array_fill

Această funcție poate fi utilizată pentru a completa cu o anumită valoare elementele unui tablou. Funcția are trei parametri: primul reprezintă indicele la care începe completarea, al doilea reprezintă numărul elementelor completate, iar cel de-al treilea indică valoarea cu care se completează elementele.

De exemplu, în urma executării instrucțiunii `$a = array_fill(5,6,'Hello World!');`, vom avea un tablou cu șase elemente (cheile acestora sunt `5`, `6`, `7`, `8`, `9` și `10`), valorile tuturor acestora fiind `Hello World!`.

Funcția array_filter

Această funcție realizează o filtrare a elementelor unui vector pe baza unei funcții. Funcția `array_filter()` are ca parametri un tablou și identificatorul funcției care este utilizată pentru filtrare.

Această funcție trebuie să fie definită și va fi apelată pentru fiecare element al vectorului. În cazul în care funcția returnează valoarea **true** (eventual, în urma conversiei la tipul `boolean`), elementul este păstrat; în caz contrar, acesta este eliminat.

Funcția `array_filter()` returnează vectorul obținut după eliminarea elementelor.

În continuare vom prezenta un exemplu în care se utilizează această funcție pentru a filtra elementele unui vector în funcție de paritatea acestora. Pagina *Web* generată este prezentată în figura 3.

```
<HTML>
  <HEAD>
    <TITLE>
      Funcția array_filter
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      function impar($var) {
        return $var % 2;
      }
```

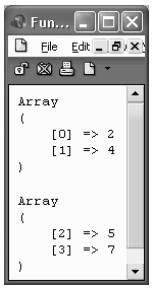

Figura 3: Funcția array_filter

```
      function par($var) {
        return 1 - $var % 2;
      }
      $vector = array("2","4","5","7");
      echo "<PRE>";
      print_r(array_filter($vector,par));
      echo "<P>";
      print_r(array_filter($vector,impar));
      echo "</PRE>";
    ?>
  </BODY>
</HTML>
```

Funcția array_flip

Această funcție are ca parametru un tablou și returnează un alt tablou în care a fost realizată o interschimbare a cheilor elementelor cu valorile. Cu alte cuvinte, pentru fiecare element va fi creat un element a cărui cheie va fi valoarea elementului în tabloul inițial și a cărui valoare va fi cheia elementului din tabloul inițial.

De exemplu, pentru tabloul definit prin

```
$tablou = array("A" => "Gazeta" , "B" => "de" ,
                                  "C" => "Informatică");
```

în urma executării apelului `$b = array_flip($tablou)` tabloul `$b` va avea forma **array**`("Gazeta" => "A" , "de" => "B" , "Informatică" => "C")`.

În cazul în care în vectorul transmis ca parametru există mai multe elemente care au aceeași valoare, datorită faptului că elementele unui vector trebuie să sibă chei distincte, vor fi eliminate toate aceste elemente, cu excepția ultimului.

Funcțiile array_intersect și array_intersect_assoc

Aceste funcții au un comportament similar funcțiilor `array_diff()`, respectiv `array_diff_assoc()`. Singura diferență este dată de faptul că tabloul rezultat va conține toate elementele care se află în fiecare dintre tablourile transmise ca parametri (pentru funcția `array_diff()` elementele trebuie să aibă aceeași valoare, în timp ce pentru funcția `array_diff_assoc()` ele trebuie să aibă aceeași cheie și aceeași valoare).

Funcția array_key_exists

Această funcție are doi parametri (o cheie și un vector) și verifică, așa cum sugerează și numele, dacă vectorul conține un element cu cheia respectivă. Funcția returnează o valoare logică, iar pentru parametrul care reprezintă cheia poate fi folosită orice valoare care poate fi cheie într-un vector.

Funcția array_keys și array_values

Aceste funcții returnează un vector care conține ca valori toate cheile (respectiv toate valorile) elementelor unui tablou transmis ca parametru. Pentru funcția `array_keys()`, în cazul în care se folosește un parametru suplimentar, vectorul rezultat va conține doar cheile elementelor care au valoarea egală cu cea indicată de acest parametru.

Funcția array_map

Această funcție realizează o transformare a elementelor unui vector pe baza unei funcții. Funcția este transmisă ca parametru și este aplicată tuturor elementelor unui vector care este, și el, transmis ca parametru.

Iată un exemplu:

```
function cub($n) {
  return $n*$n*$n;
}
$a = array(1, 2, 3, 4, 5);
$b = array_map("cub", $a);
```

În urma executării secvenței, vectorul `$b` va avea cinci elemente ale căror valori vor fi `1`, `8`, `27`, `64` și `125`.

Există posibilitatea de a transmite mai multe tablouri ca parametri. În acest caz, numărul argumentelor funcției utilizate trebuie să fie egal cu numărul tablourilor transmise și se recomandă ca tablourile să aibă același număr de elemente. În caz contrar, tablourile mai "scurte" vor fi completate cu elemente nule.

Un artificiu interesant permite crearea unor vectori de vectori folosind funcția `array_map()`. Pentru aceasta vom utiliza valoarea **`null`** pentru funcția transmisă ca parametru. Vom prezenta în continuare un exemplu:

```
$a = array("unu","doi","trei");
$b = array("one","two","three");
$c = array("un","deux","trois");
$d = array_map(null,$a,$b,$c);
```

Vectorul `$d` va avea următoarea structură:

```
Array
(
    [0] => Array
        (
            [0] => unu
            [1] => one
            [2] => un
        )
    [1] => Array
        (
            [0] => doi
            [1] => two
            [2] => deux
        )
    [2] => Array
        (
            [0] => trei
            [1] => three
            [2] => trois
        )
)
```

Funcțiile array_merge și array_merge_recursive

Aceste funcții realizează o "interclasare" a două sau mai multe tablouri prin adăugarea elementelor unui vector la sfârșitul tabloului anterior. În cazul în care există elemente cu aceeași cheie (reprezentată de un șir de caractere) în tabloul rezultat va apărea un singur element cu cheia respectivă, și anume primul întâlnit în timpul interclasării. În cazul în care cheia identică este reprezentată de un număr, în tabloul rezultat vor apărea ambele valori, dar valoarea corespunzătoare unei chei care apare deja în tablou va avea

cheia reprezentată de cel mai mic număr mai mare decât cheia inițială care nu apare încă în tabloul rezultat.

În cazul în care tablourile conțin elemente care sunt tablouri, atunci funcția este `array_merge_recursive()` și are ca efect și interclasarea acestor tablouri (care sunt elemente ale altor tablouri) și procedeul continuă recursiv. În cele ce urmează vom prezenta un document *PHP* care ilustrează diferențele dintre cele două funcții. Efectul este prezentat în figura 4.

```
<HTML>
  <HEAD>
    <TITLE>
      Funcțiile
      array_merge și
      array_merge
      _recursive
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      $v1 = array(
      "culoare" =>
      array(
      "preferată" =>
      "roșu"),(5);
      $v2 = array(
      10,("culoare"
      =>(array(
      "preferată" =>
      "verde",
      "albastru"));
      echo "<PRE>";
      print_r(array_merge($v1,$v2));
      echo "<P>";
      print_r(array_merge_recursive($v1,$v2));
      echo "</PRE>";
    ?>
  </BODY>
</HTML>
```

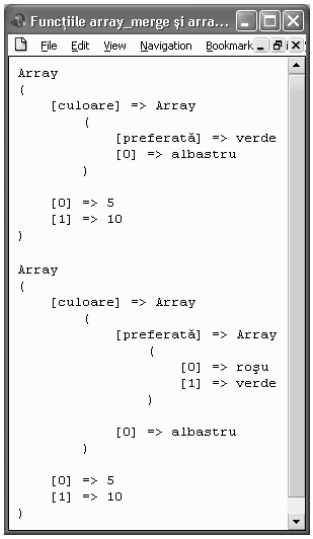

Figura 4: Funcțiile array_merge și array_merge_recursive

Funcția array_multisort

Această funcție poate fi utilizată pentru a sorta mai multe tablouri sau un tablou multidimensional.

Primul parametru al funcției trebuie să fie un tablou, iar următoarele pot fi tablouri și indicatori de sortare. Fiecare indicator de sortare descrie caracteristici ale sortării pentru cel mai apropiat tablou aflat înaintea indicatorului în lista de parametri.

Există două categorii de indicatori de sortare:

- indicatori de ordine:
  - `SORT_ASC`: sortare în ordine crescătoare;
  - `SORT_DESC`: sortare în ordine descrescătoare;
- indicatori de tip :
  - `SORT_REGULAR`: elementele sunt comparate fără a efectua conversii;
  - `SORT_NUMERIC`: elementele sunt comparate după ce sunt convertite la valori numerice;
  - `SORT_STRING`: elementele sunt comparate după ce sunt convertite în șiruri de caractere.

Pentru un tablou nu pot fi specificați doi indicatori care fac parte din aceeași categorie (de exemplu, nu este permisă specificarea pentru un tablou a indicatorilor `SORT_NUMERIC` și `SORT_STRING`).

Nu este obligatorie specificarea ambelor tipuri de indicatori pentru fiecare tablou. În cazul în care indicatorii lipsesc, sunt utilizate valorile implicite `SORT_ASC` și `SORT_REGULAR`.

Tablourile transmise ca parametri sunt considerate a fi coloane într-o matrice care trebuie sortată pe linii. Primul tablou reprezintă criteriul de bază. Pentru valori egale în primul tablou se ia în considerare al doilea criteriu (reprezentat de al doilea tablou). Pentru valori egale în primele două tablouri se ia în considerare al treilea criteriu și așa mai departe.

Funcția returnează o valoare logică ce indică dacă operația de sortare s-a încheiat cu succes.

Să considerăm doi vectori definiți astfel:

```
$v1 = array("10",100,100,"a");
$v2 = array(1,3,"2",1);
```

În urma execuției apelului `array_multisort($v1, $v2);` vectorul `$v1` va conține elementele `"10"`, `"a"`, `100` și `100`, în timp ce vectorul `$v2` va conține elementele `1`, `1`, `"2"` și `3`. Se observă că elementele din al doilea vector corespunzătoare elementelor cu valori egale din primul vector (`3` și `"2"` corespund celor două elemente cu valoarea `100` din primul vector) au fost și ele sortate.

Vom prezenta acum un exemplu în care vom sorta un tablou bidimensional definit prin:

```
$v = array(array("10",100,100,"a"),
           array(1,3,"2",1));
```

În urma unui apel de forma:

```
array_multisort($v[0],SORT_ASC,SORT_STRING,
                $v[1],SORT_NUMERIC,SORT_DESC);
```

prima linie a tabloului va conține elementele `"10"`, `100`, `100` și `"a"` (au fost sortate crescător ca șiruri de caractere), iar cea de-a doua va conține elementele `1`, `3`, `"2"` și `1` (pentru valori egale în prima linie, elementele de pe a doua au fost sortate descrescător ca valori numerice).

Funcția array_multisort

Această funcție poate fi utilizată pentru a completa un tablou cu o valoare dată, astfel încât să aibă o dimensiune specificată.

Funcția are trei parametri: tabloul care trebuie completat, dimensiunea pe care trebuie să o aibă tabloul completat și valoarea cu care este completat tabloul.

Tabloul va fi completat astfel încât dimensiunea sa să fie egală cu valoarea absolută a celui de-al doilea parametru. În cazul în care valoarea acestui parametru este pozitivă, tabloul va fi completat la dreapta, iar dacă aceasta este

negativă, tabloul va fi completat la stânga. Dacă valoarea absolută a acestui parametru este mai mică sau egală cu dimensiunea tabloului transmis ca parametru, atunci nu sunt adăugate elemente. Funcția returnează tabloul obținut după completare.

Să considerăm vectorul definit prin:

```
$v = array(2,4,5,7);
```

În continuare vom prezenta câteva apeluri ale funcției și vectorii returnați de aceasta:

- `$r = array_pad($v2,6,-1): (2,4,5,7,-1,-1);`
- `$r = array_pad($v2,-7,0): (0,0,0,2,4,5,7);`
- `$r = array_pad($v2,3,5): (2,4,5,7);`

[*nu este adăugat nici un element*]

Funcțiile array_pop și array_push

Aceste funcții permit utilizarea tablourilor similar stivelor. Funcția `array_pop()` este utilizată pentru a elimina un element al tabloului (ultimul), iar funcția `array_push()` pentru a adăuga în tablou unul sau mai multe elemente.

Să considerăm că vectorul `$v` este vid. În urma execuției apelului `array_push($v,1,2,3)` acesta va conține elementele `1`, `2` și `3`. În acest moment, un apel de forma `array_pop($v)` va returna valoarea `3`, iar vectorul `$v` va conține elementele `1` și `2`.

În cazul în care vectorul transmis ca parametru este vid, funcția `array_pop()` returnează valoarea **`null`**. Funcția `array_push()` va returna întotdeauna noua dimensiune a vectorului.

Funcțiile array_shift și array_unshift

Perechile funcțiilor `array_pop()` și `array_push()` sunt `array_shift()` și `array_unshift()`. Acestea pot fi utilizate pentru a insera sau elimina elemente la începutul unui tablou. Executarea acestor funcții va duce la modificarea cheilor întregi astfel încât acestea să înceapă de la `0`.

Funcția array_rand

Această funcție permite alegerea aleatoare a unui număr de elemente dintr-un vector.

Funcția are ca parametri un vector și o valoare numerică ce indică numărul elementelor care vor fi alese. Acest al doilea parametru poate lipsi, caz în care are valoarea implicită `1`.

În cazul în care se alege un singur element, funcția va returna cheia acestuia. În caz contrar, funcția va returna un tablou care va conține cheile elementelor alese.

În continuare vom prezenta un exemplu care ilustrează modul de utilizare a acestei funcții. Efectul este prezentat în figura 5.

```
<HTML>
  <HEAD>
    <TITLE>
      Funcția array_rand
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      $v =
       array(2,4,5,7);
      $a =
       array_rand($v);
      echo "A fost
            ales
            elementul
            a cărui
            cheie este
            <B>$a</B>. Valoarea acestui
            element este
            <B>$v[$a]</B>.<BR><BR>";
      $a = array_rand($v);
      echo "A fost ales elementul a cărui
            cheie este <B>$a</B>. Valoarea
            acestui element este
            <B>$v[$a]</B>.<BR><BR>";
      $a = array_rand($v,2);
      echo "Au fost alese elementele ale
            căror chei sunt <B>$a[0]</B> și
            <B>$a[1]</B>. Valorile acestor
            elemente sunt <B>".$v[$a[0]].
            "</B> și <B>".$v[$a[1]].
            "</B>.<BR><BR>";
      $a = array_rand($v,2);
      echo "Au fost alese elementele ale
            căror chei sunt <B>$a[0]</B> și
            <B>$a[1]</B>. Valorile acestor
            elemente sunt <B>".$v[$a[0]].
            "</B> și <B>".$v[$a[1]]."</B>.";
    ?>
  </BODY>
</HTML>
```

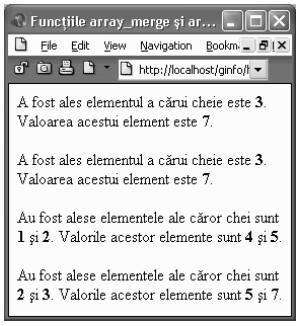

Figura 5: Funcția array_rand

Funcția array_reduce

Această funcție aplică o altă funcție transmisă ca parametru asupra elementelor unui tablou până în momentul în care tabloul este redus la o singură valoare.

Mai exact, funcția transmisă ca parametru este apelată pentru primele două elemente ale tabloului. Cele două elemente sunt eliminate și în tablou este inserat pe prima poziție rezultatul apelului funcției. Procedeul continuă până în momentul în care vectorul conține o singură valoare care este retunată.

Există și posibilitatea de a transmite o valoare inițială pentru efectuarea operațiilor. În acest caz funcția transmisă ca parametru va fi apelată pentru valoarea inițială și primul element al tabloului. În cazul în care tabloul este vid, rezultatul apelului va fi această valoare inițială.

Să considerăm funcțiile definite astfel:

```
function S($v,$w){
  return $v + $w;
}
```

```
function P($v,$w){
  return $v + $w;
}
```

În urma executării instrucțiunilor:

```
$a = array(2,4,5,7);
$x = array();
$b = array_reduce($a,"S");
$c = array_reduce($a,"P",10);
$d = array_reduce($x,"S",1);
```

valorile variabilelor `$b`, `$c` și `$d` vor fi `18` (`2+4+5+7`), `2800` (`10*2*4*5*7`), respectiv `1` (datorită faptului că vectorul `$x` este vid).

Funcția array_reverse

Această funcție inversează ordinea elementelor unui vector transmis ca parametru. În cazul în care este prezent un al doilea parametru și valoarea acestuia este **TRUE** (eventual după o conversie), atunci se păstrează corespondența dintre chei și valori. Funcția returnează tabloul obținut după inversare.

Să considerăm tabloul definit astfel:

```
$v = array("Gazeta","de","Informatică",
                          array("mai",2003));
```

În urma apelurilor `$a = array_reverse($v)` și `$b = array_reverse($v,TRUE)` tablourile `$a` și `$b` vor avea forma prezentată în figura 6.

```
Array                          Array
(                              (
    [0] => Array                   [3] => Array
        (                              (
            [0] => mai                     [0] => mai
            [1] => 2003                    [1] => 2003
        )                              )
    [1] => Informatică             [2] => Informatică
    [2] => de                      [1] => de
    [3] => Gazeta                  [0] => Gazeta
)                              )
```

Tabloul $a — Tabloul $b

Figura 6: Efectul funcției array_reverse

În figură se observă clar diferențele dintre cheile celor două tablouri.

Funcția array_search

Această funcție verifică dacă un vector conține un element care are o anumită valoare și, dacă un astfel de element există, returnează cheia acestuia. În caz contrar funcția returnează valoarea **FALSE**. Primul parametru al funcției este valoarea căutată, iar al doilea este vectorul în care este căutată valoarea respectivă. Funcția permite folosirea unui al treilea parametru care este opțional. În cazul în care acesta există și are valoarea **TRUE**, atunci în timpul căutării se verifică și egalitatea tipurilor.

Funcția array_slice

Această funcție poate fi utilizată pentru a prelua o porțiune a unui tablou. Primul parametru reprezintă tabloul, al doilea reprezintă poziția la care începe porțiunea preluată, iar al treilea indică numărul elementelor preluate.

Această funcție ignoră cheile elementelor și ia în considerare pozițiile "reale" ale elementelor în tablou.

În cazul în care valoarea celui de-al doilea parametru este pozitivă, poziția de început este determinată relativ față de începutul tabloului. Pentru o valoare negativă, poziția este determinată relativ față de sfârșitul tabloului.

În cazul în care valoarea celui de-al treilea parametru este pozitivă, ea va indica numărul elementelor preluate. Dacă ea este negativă, atunci valoarea sa absolută va indica poziția față de sfârșitul vectorului la care se va "opri" preluarea elementelor. Acest parametru poate lipsi, caz în care sunt preluate toate elementele, până la sfârșitul tabloului.

Funcția va returna un tablou care va conține elementele preluate.

Vom prezenta în continuare modul în care poate fi utilizată această funcție pentru a prelua porțiuni ale unui vector. Efectul este ilustrat în figura 7.

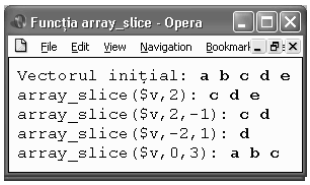

Figura 7: Funcția array_slice

```
<HTML>
  <HEAD>
    <TITLE>
      Funcția
          array_slice
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      function DisplayArray($v){
        echo "<B>";
        foreach ($v as $val)
          echo " $val";
        echo "</B><BR>";
      }
      $v = array("a","b","c","d","e");
      echo "<TT>Vectorul inițial:";
      DisplayArray($v);
      echo "array_slice(\$v,2):";
      DisplayArray(array_slice($v,2));
      echo "array_slice(\$v,2,-1):";
      DisplayArray(array_slice($v,2,-1));
      echo "array_slice(\$v,-2,1):";
      DisplayArray(array_slice($v,-2,1));
      echo "array_slice(\$v,0,3):";
      DisplayArray(array_slice($v,0,3));
      echo "</TT>";
    ?>
  </BODY>
</HTML>
```

Funcția array_splice

Efectul acestei funcții este similar efectului funcției `array_slice()`, diferența constând în faptul că porțiunea preluată este eliminată din tablou și înlocuită cu elementele unui al patrulea parametru care reprezintă un vector.

Funcția array_sum

Această funcție calculează suma elementelor unui tablou. În funcție de tipul elementelor, rezultatul va fi un număr

întreg sau un număr real. Pentru valorile nenumerice se încearcă efectuare unei conversii la numere întregi sau reale și în cazul în care valorile pot fi convertite (rezultatul conversiei nu este 0) ele sunt luate în considerare în momentul calculării sumei. De exemplu, în urma executării secvenței:

```
$a = array(2,4,6,8);
echo "suma(a) = ".array_sum($a)."\n";
$b = array("a"=>1.2,"b"=>2.3,"c"=>3.4);
echo "sum(b) = ".array_sum($b)."\n";
```

se va afișa:

```
sum(a) = 20
sum(b) = 6.9
```

Funcția array_unique

Această funcție elimină dintr-un vector toate valorile care apar de două sau mai multe ori și returnează vectorul obținut în urma eliminării.

Pentru a elimina elementele, funcția le sortează mai întâi considerându-le șiruri de caractere. Apoi, pentru fiecare valoare care apare de mai multe ori, va păstra prima cheie întâlnită în șirul sortat.

Funcții de sortare

Interpretorul *PHP* pune la dispoziție o mulțime de funcții care pot fi utilizate pentru a sorta elementele unui tablou. Acestea sunt (în ordine alfabetică):

- `arsort()` - sortează un tablou în ordine descrescătoare și păstrează asocierile dintre indici;
- `asort()` - sortează un tablou în ordine crescătoare și păstrează asocierile dintre indici;
- `krsort()` - sortează un tablou în ordine descrescătoare în funcție de cheile elementelor acestuia;
- `ksort()` - sortează un tablou în ordine crescătoare în funcție de cheile elementelor acestuia;
- `natcasesort()` - sortează un tablou folosind o "ordine naturală" fără a face distincție între literele mari și literele mici;
- `natsort()` - sortează un tablou folosind o "ordine naturală" făcând distincție între literele mari și literele mici
- `rsort()` - sortează un tablou în ordine descrescătoare în funcție de valorile elementelor acestuia;
- `sort()` - sortează un tablou în ordine crescătoare în funcție de valorile elementelor acestuia;
- `uasort()` - sortează un tablou folosind pentru comparări o funcție indicată de utilizator și păstrează asocierile dintre indici;
- `uksort()` - sortează un tablou în funcție de cheile acestuia folosind pentru comparări o funcție indicată de utilizator;
- `usort()` - sortează un tablou folosind pentru comparări o funcție indicată de utilizator.

Toate funcțiile necesită un parametru care reprezintă vectorul care trebuie sortat.

Funcțiile `arsort()`, `asort()`, `krsort()`, `ksort()`, `rsort()` și `sort()` permit utilizarea unui al doilea parametru (opțional) care este un indicator de tip. Valoarea implicită a acestuia este `SORT_REGULAR`. Al doilea parametru al funcțiilor `uasort()`, `uksort()` și `usort()` reprezintă identificatorul funcției care va fi folosită pentru comparare.

Funcțiile `natsort()` și `natcasesort()` nu permit utilizarea unui alt parametru și stabilesc ordinea așa cum ar face o ființă umană (ordine naturală).

De exemplu, pentru șirurile de caractere `IMG1.GIF`, `IMG2.GIF`, `IMG10.GIF` și `IMG12.GIF`, un algoritm "clasic" (bazat pe ordinea lexicografică) ar stabili ordinea: `IMG1.GIF`, `IMG10.GIF`, `IMG12.GIF` și `IMG2.GIF`, dar algoritmul care folosește ordinea naturală nu ar efectua nici o modificare.

Funcția compact

Această funcție are ca argumente un număr variabil de șiruri de caractere sau tablouri. Ea creează un tablou în care va insera toate șirurile care reprezintă o variabilă (aceste șiruri vor fi chei, iar valorile elementelor corespunzătoare vor fi valorile variabilelor). În cazul tablourilor, vor fi luate în considerare toate elementele care sunt șiruri de caractere, iar dacă tablourile conțin elemente care, la rândul lor, sunt tablouri, atunci procedeul se aplică recursiv și asupra acestora.

Funcțiile count și sizeof

Aceaste funcții returnează numărul elementelor unui vector transmis ca parametru. Teoretic, parametrul poate să nu fie un vector, dar în acest caz se va returna valoarea `1` (doar tablourile pot conține mai multe elemente). Nu există nici o diferență între aceste funcții.

Funcții pentru elementul curent

Fiecărui vector îi este asociată o referință care indică elementul curent. Există mai multe funcții care pot utiliza această referință. Acestea sunt (în ordine alfabetică):

- `current()` - returnează valoarea elementului curent al tabloului;
- `each()` - returnează cheia și valoarea elementului curent și deplasează referința la elementul următor; funcția returnează un tablou cu patru elemente: două dintre ele au cheile `0` și `key` și conțin cheia elementului; celelalte două au cheile `1` și `value` și conțin valoarea elementului;
- `end()` - deplasează referința la ultimul element al tabloului;
- `key()` - returnează cheia elementului curent al tabloului;
- `next()` - deplasează referința la următorul element al tabloului;
- `pos()` - returnează valoarea elementului curent al tabloului (are același efect ca și funcția `current()`);
- `prev()` - deplasează referința la elementul anterior al tabloului;
- `reset()` - deplasează referința la primul element al tabloului.

Funcția extract

Această funcție este utilizată pentru a crea variabile pe baza valorilor elementelor unui vector. Cheile elementelor vor deveni identificatori de variabile, iar valorile elementelor vor deveni valorile variabilelor respective.

Funcția verifică dacă cheile reprezintă identificatori valizi ai variabilelor sau dacă se încearcă crearea unei variabile care există deja. Aceste situații sunt tratate în funcție de valoarea unui al doilea parametru care reprezintă un indicator și este opțional. Valorile posibile pentru acest parametru sunt următoarele:

- `EXTR_OVERWRITE`: dacă variabila există deja, valoarea curentă va fi suprascrisă;
- `EXTR_OVERWRITE`: dacă variabila există deja, valoarea curentă nu va fi modificată;
- `EXTR_PREFIX_SAME`: dacă variabila există deja, denumirea noii variabile va fi prefixată cu un prefix transmis ca al treilea parametru al funcției;
- `EXTR_PREFIX_ALL`: denumirile tuturor variabilelor vor fi prefixate;
- `EXTR_PREFIX_INVALID`: în cazul în care o cheie nu reprezintă un identificator de variabilă validă, denumirea cheii va fi prefixată;
- `EXTR_IF_EXISTS`: nu sunt create variabile noi, ci sunt doar suprascrise valorile variabilelor care există deja;
- `EXTR_PREFIX_IF_EXISTS`: în cazul în care cheia reprezintă denumirea unei variabile, se creează o nouă variabilă prefixată; dacă nu există o astfel de variabilă, nu este creată o variabilă nouă;
- `EXTR_REFS`: variabilele create reprezintă referințe; așadar, prin intermediul lor pot fi modificate și elementele vectorului; acest indicator poate fi utilizat împreună cu oricare dintre celelalte cu ajutorul unei disjuncții logice.

Cel de-al treilea parametru al funcției trebuie să apară doar dacă valoarea celui de-al doilea este unul din indicatorii ai cărui identificator conține secvența `PREFIX`.

Al doilea parametru poate lipsi, caz în care are valoarea implicită `EXTR_OVERWRITE`.

Funcția in_array

Această funcție este similară funcției `array_search()`, singura diferență fiind faptul că, în cazul în care căutarea se încheie cu succes, nu este returnată cheia elementului găsit, ci valoarea `TRUE`.

Funcția range

Această funcție returnează un vector care conține toate valorile cuprinse între două limite date (`st` și `dr`). Primul element al vectorului va avea valoarea `st`, iar următoarele vor avea valorile `st + 1`, `st + 2` etc. dacă `st < dr`, respectiv `st - 1`, `st - 2` etc. dacă `st > dr`. Începând cu versiunea *PHP* 4.4.0 va exista și posibilitatea de a crea șiruri în care diferența dintre două elemente să fie diferită de `1`.

În cele ce urmează vom prezenta un document *PHP* care ilustrează modul de utilizare a acestei funcții. Efectul este ilustrat în figura 8.

```
<HTML>
  <HEAD>
    <TITLE>
      Funcția range
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      function DisplayArray($v){
        echo "<B>";
        foreach ($v as $val)
          echo " $val";
        echo "</B><BR>";
      }
      echo "<TT>";
      DisplayArray(range(1,10));
      DisplayArray(range(10,1));
      DisplayArray(range('a','k'));
      DisplayArray(range('k','a'));
      echo "</TT>";
    ?>
  </BODY>
</HTML>
```

Figura 8: Funcția range

Funcția shuffle

Această funcție poate fi utilizată pentru a "amesteca" elementele unui tablou. Ea nu returnează nici o valoare, dar modifică ordinea elementelor tabloului.

Vom prezenta în continuare un document *PHP* care conține un exemplu de utilizare a acestei funcții, efectul fiind ilustrat în figura 9.

```
<HTML>
  <HEAD>
    <TITLE>
      Funcția shuffle
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      function DisplayArray($v){
        echo "<B>";
        foreach ($v as $val)
          echo " $val";
        echo "</B><BR>";
      }
      echo "<TT>Șirul inițial:<BR> ";
      $a = range(1,10);
      DisplayArray($a);
      echo "Șirul amestecat:<BR> ";
      shuffle($a);
      DisplayArray($a);
      echo "</TT>";
    ?>
  </BODY>
</HTML>
```

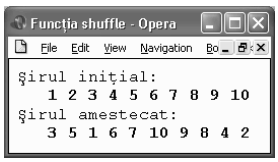

Figura 9: Funcția shuffle