Funkcijos

Funkcijos yra būdas grupuoti bloką fukcijų nevienkartiniam paprastam naudojimui.

Funkcijos programavime yra panašios į matematines funkcijas (pvz.: sin(x) - duos tam tikrą skaičių priklausomai nuo rezultato). Funkcijos pagreitina darbą.

Norėdami surasti pilną PHP funkcijų sarašą, užeikite į PHP interneto puslapį http://www.php.net/manual/en/funcref.php.

Parametras yra funkcijos apibrėžimo kintamasis.

Kai iškviečiama funkcija, argumentai yra duomenys, kuriuos perduodate funkcijos parametrams.

Funkcijos deklaracijoje parametras yra kintamas.

Argumentas yra tikroji šio kintamojo, kuris perduodamas funkcijai, vertė.

Sintaksė:

```
function funkcijos_vardas([parameterai])
{
global $var;
// kodas
      return true;
}
```

Savybės:

a) Raktinis žodis `function` pasako PHP kad jūs deklaruojante funkciją.
b) Kiekviena funkcija turi savo vardą (funkcijos_vardas).
c) Funkcijos turi parametrus, jei juos nustatote (gali ir nebūti).
d) Return gražina kokį nors kintamajį ar panašiai. Jo nebūtina rašyti. Tuo atveju funkcija gražins NULL.

Pavyzdžiai:

```
function sudeti($vienas, $du)
```

```
{
    $rezultatas = $vienas + $du;
    return $rezultatas;
}

//Norint iškviesti šią funkciją, reikia paduoti du parametrus:
echo sudeti(1,5);
```

`$vienas` ir `$du` yra parameterai, kintamieji, kurie egzistuoja tiktai pačioje funkcijoje. Juos galima paduoti kaip vidinius kintamuosius arba kaip nuorodas..
Parametrams iš anksto galima suteikti reikšmes ir kviečiant funkciją juos praleisti.

```
function sudeti($vienas = 10, $du = 15)
{
    return $vienas + $du;
}
echo sudeti().'<br>';
echo sudeti(1).'<br>';
echo sudeti(1,2).'<br>';
```

Galime sukurti funkciją, kurioje argumentų skaičius yra neribotas

```
function vidurkis(...$skaiciai)
{
    $sudetis = 0;
    foreach ($skaiciai as $val)
  {
    $sudetis += $val;
  }
  $vidurkis = $sudetis / count($skaiciai);
  return $vidurkis;
}
echo vidurkis(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20);
```

Funkcijos viduje ir išorėje esantys kintamieji, pavadinti tuo pačiu vardu yra skirtingi kintamieji:

```
$kintamasis = 'Labas';
function labas()
{
    echo $kintamasis;
}

labas();
```

Kodas rodys klaidą, nes kintamasis funkcijos viduje neaprašytas.
Jeigu norime, kad išorėje esantis kintamasis būtų matomas ir viduje jį turime funkcijoje aprašyti:

```
$kintamasis = 'Labas';
function labas()
{
    global $kintamasis;
    echo $kintamasis;
}
labas();
```

Funkcijose gali būti naudojami statiniai kintamieji:

```
   function foo() {
       static $index = 0;
       $index++;
       echo "$index\n";
   }
```

Kviečiant `foo()` kelis kartus:

```
foo();
foo();
foo();
```

Funkcija išves rezultatus:

1
2
3

Rekursinė funkcija, tai tokia funkcija, kuri savo viduje kviečia pati save:

```php
function recursive($num){
    echo $num, '<br>';
    if($num < 50){
        //Kviečiame save. Padidiname numerį vienetu.
        return recursive($num + 1);
    }
}
$startNum = 1;
recursive($startNum);
```

Anoniminės funkcijos, taip pat žinomos kaip ***`closures`***, leidžia sukurti funkcijas, kurios neturi vardo. Jos yra naudingiausios kaip `callback` parametrai kitose funkcijose, tačiau jos turi ir daug kitų panaudojimo galimybių.
Kaip `callback` parametras:

```php
$masyvas = [
    ['a','d'],
    ['v','e'],
    ['a','c'],
    ['s','r'],
];
usort($masyvas, function($a, $b){
    return $a[0] <=> $b[0];
});
```

Anoniminė funkcija priskirta kintamajam:

```
$greet = function($name)
{
    printf("Hello %s", $name);
};
$greet('World');
$greet('PHP');
```

Anoniminė funkcija ir matomumo ribos:

```
$result = 0;
$one = function()
{ var_dump($result); };

$two = function() use ($result)
{ var_dump($result); };

$three = function() use (&$result)
{ var_dump($result); };

$result++;

$one();     // NULL: $result nepasiekiamas
$two();     // int(0): $result nukopijuojamas
$three();     // int(1)
```

Anoniminė rekursinė funkcija:

```
 $func = function ($limit = NULL) use (&$func) {
   static $current = 10;

   // tikrinam eigą
   if ($current <= 0) {
```

```
        //išeinam
        return FALSE;
    }

    // spausdinam reikšmę.
    echo "$current<br>";

    $current--;

    $func();
};
 //  Kviečiam funkciją
$func();
```