Php(3)

structuri conditionale

Instructiunea IF

```
<?php
$a = 12;
$b = 8;
$rezultat = $a + $b;
if($rezultat == 20) {
echo “Rezultatul este perfect”;
}
?>
```

Instructiunea ELSE

```
<?php
$a = 20;
$b = 8;
$rezultat = $a + $b;
if($rezultat == 20) {
echo “Rezultatul este perfect”;
} else {
echo “altceva incorect”;
}
?>
```

- **Instructiunea ELSEIF**

- **<?php**
- $a = 20;
- $b = 1;
- $rezultat = $a + $b;
- **if**($rezultat == '20') {
- echo 'Rezultatul este egal cu 20';
- } **elseif** ($rezultat == '21') {
- echo 'Rezultatul este egal cu 21';
- } **else** {
- echo 'Rezultatul nu este egal cu cel din conditie';
- }
- **?>**

- **Instructiunea SWITCH**
- Aceasta instructiune functioneaza asemanator cu cea if, insa permite conditiilor sa aibe mai mult de 2 valori.
- Intr-o instructiune if, conditia poate fi adevarata sau falsa, insa intr-o instructiune **switch** conditia poate lua
- orice numar de valori diferite.
- Aceasta instructiune trebuie sa contina o instructiune **case** care sa manevreze fiecare valoare pe care o doriti.
- **<?php**
- **if**(!isset($_GET['modul'])) $_GET['modul'] = '';
- **switch**($_GET['modul']) {
- **case** '':
- echo 'Pagina switch.php';
- **break**;
- **case** 'pagina1':
- echo 'Pagina switch.php?modul=pagina1';
- **break**;
- **case** 'pagina2':
- echo 'Pagina switch.php?modul=pagina2';
- **break**;
- }
- **?>**

Instrucţiuni de ciclare

Bucla WHILE

.

```
<?php
$numar = 1;
while($numar <= 5){
echo $numar.”<br />”;
$numar++;
}
?>
```

Structura FOR

for(expresie1; conditie; expresie2) {
//instructiune
}
Oricare dintre cele trei expresii poate lipsi; in cazul in care o expresie lipseste, se considera ca ea are valoarea
true.

```
<?php
for ($i = 1; $i <= 10; $ia++) {
echo “Nota: .”$variabila.”<br />”;
}
?>
```

Structura FOREACH

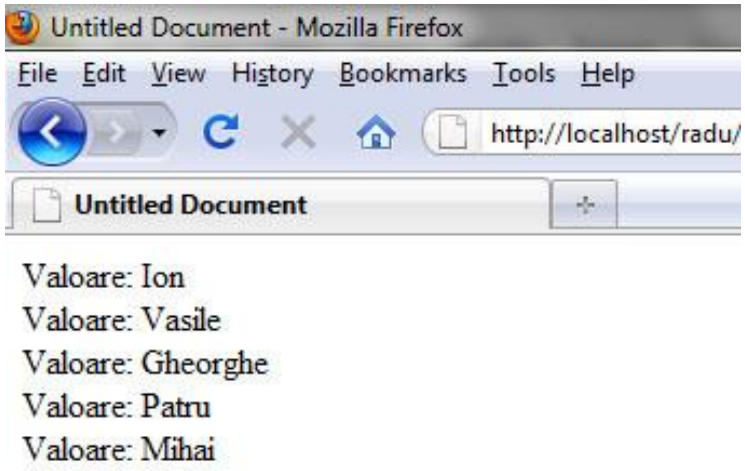
Aceasta structura poate fi folosita pentru a realiza o repetare printre toate elementele unui vector.
Exista doua sintaxe acceptate pentru aceasta structura si anume:

foreach(expresie_vectoriala **as** $valoare) {
//instructiune
}
la fiecare iteratie valoarea elementului curent este atribuita variabilei *$valoare,* si apoi se trece la elementul urmator

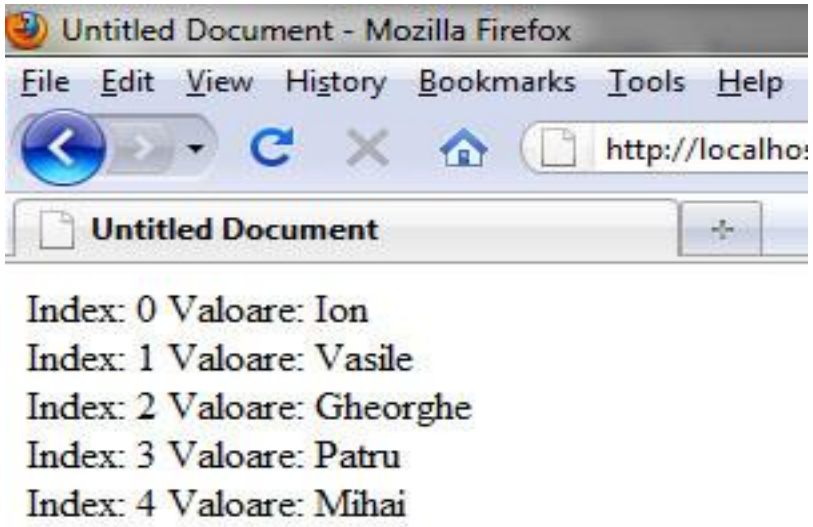
foreach(expresie_vectoriala **as** $cheie => $valoare) {
//instructiune
}
la fiecare iteratie valoarea indexului elementului curent este atribuita variabilei *$cheie*.

Exemplu **foreach**.

```
<?php

$sir = array("Ion", "Vasile", "Gheorghe", "Patru", "Mihai");

foreach($sir as $valoare) {
echo "Valoare: ".$valoare." <br />\n";
}
?>
```



```php
<?php

$sir = array("Ion", "Vasile", "Gheorghe", "Patru", "Mihai");

foreach($sir as $index => $valoare) {
echo "Index: ".$index." Valoare: ".$valoare." <br />\n";
}
?>
```



Instructiunea BREAK

break

intrerupe fortat executia unui ciclu sau a secventei de instructiuni
argumente: valoare implicita 1 – se intrerupe executia unei singure structuri.

```
foreach ($a as $v){
if($v < 0)
break;}

for($i = 0; $i < $n; $i++)
for($j = 0; $j < $n; $j++)
if(!$a[$i][$j])
break 2;
```

Instructiunea **break** intrerupe: **for**, **foreach**, **while**, **do - while** si **switch**.

Instructiunea CONTINUE

continue

intrerupe executia secventei de instructiuni din interiorul unui ciclu si trecerea la urmatoarea iteratie.
la **for**, inainte de urmatoarea iteratie se evalueaza (executa) expresia de incrementare

Exemplul : afisarea elementelor unui sir de numere intregi care sunt mai mari decat 500.

```
foreach($a as $v) {
if($v <= 500)
continue;
echo $v;
}
```

Exemplu2: Folosirii argumentelor pentru instructiunea **continue**.

```
<?php
$i = 0;
while($i++ < 5) {
        echo "Ciclul #1 <br />\n";
                while(1) {
                        echo "  Ciclul #2 <br />\n";
                        while (1) {
                                echo "  Ciclul #3<br />\n";
                                continue 3;
                                }
                        echo “Va fi afisat?.<br />\n";
                        }
                echo “Dar acesta?.<br />\n";
                }
?>
```

Alte structuri PHP

Structurile
include, **require**, **include_once** si **require_once**
pot fi utilizate pentru a "insera" anumite instructiuni care sunt pastrate intr-un alt fisier (document).
Interpretorul PHP considera ca secventa din fisierul inserat se afla in fisierul din care s-a apelat inserarea in pozitia in care apare structura de inserare.

Clase si obiecte

Ce este o clasa ?

O clasa este o colectie de variabile si functii care opereaza asupra variabilelor respective.
Sintaxa in PHP este:

```
<?php
class nume_clasa {
// date membre
var nume_variabila_1
// ...
var nume_variabila_m*
// metode
function nume_functie_1 (parametri) {
// definitia functiei
}
// ...
function nume_functie_n (parametri) {
// definirea functiei
}
}
?>
```

Numele unei clase nu poate: sdtclass
In PHP, datele membre nu pot fi initializate decat cu valori constante.

Pentru a initializa variabilele cu valori care nu sunt constante trebuie folosit un constructor.
Contraexemplu: *:*

```
class gresit {
var $data = date ('Y-m-d');
var $nume = $prenume;
var $dest = “pop” . “ion”;
var $obiect = array (“nimic”, “si nimic”);
}
```

Obiectele

In PHP clasele sunt considerate a fi tipuri de date;

Pentru a crea o variabila al carei tip este o clasa, trebuie utilizat operatorul **new**. Exemplu: clasa Mate cu doua date membre x si y care sunt numere intregi si doua metode care realizeaza adunarea, respectiv inmultirea lor.

```
class Mate {
var x = 1;
var y = 2;
function Sum() {
return $this -> x + $this -> y;
}
function Produs() {
return $this -> x * $this -> y;
}
}
```

Pentru a crea un obiect de tipul Mate, vom utiliza o instructiune de tipul:

```
$aritm = new Mate;

 //apelam cele doua metode astfel:

echo $aritm -> Sum();
echo $aritm -> Produs();

//Valorile datelor membre pot fi modificate:
$aritm -> x = 23;
```

Exemplu

```
<?php
class Salutare {
var $salut = “Salut toti chinuitii!”
function Salut() {
return $this -> salut;
}
}
$salutul = new Salutare;
echo $salutul -> Salut().”<br />”;
?>
```