Informatică Aplicată 1 – Îndrumar de Laborator

Capitolul 2 – Introducere în PHP

1) Elemente de bază

PHP-ul este un limbaj de scripting pentru Web utilizat pentru a crea pagini Web interactive, dinamice, cu funcționalități extinse (baze de date, e-mail, multimedia etc.). Ca structură PHP-ul este foarte apropiat de C. Totuși, după cum se afirmă în [BM], PHP-ul este un limbaj de programare comod pentru începători, chiar și pentru cei care nu au mai programat în trecut.

PHP este un limbaj pentru inserturi în fișiere HTML, inserturi care sunt înlocuite la evaluarea fișierului de către serverul de Web cu rezultatele execuției lor [FR]. Serverul se comportă ca un interpretor și nu ca un compilator.

Convenții pentru numele fișierelor PHP

Fișierele PHP **trebuie** să aibă doar extensia **.php** , iar numele fișierului nu trebuie să conțină spații. Pentru a evita problemele, formați numele de fișiere doar din litere mici, cifre, sau caracterul - (minus) [BM]. Exemple: index.php, script125.php, fisierul-meu.php

Cel mai simplu script PHP

Creați în folderul ***htdocs*** un nou fișier text, alegeți un nume și apoi schimbați-i extensia în .php

Cel mai simplu script PHP, care afișează textul „Hello world!" este următorul:

```
<?php
  echo("Hello world!");
?>
```

Marcajele **<?php** … **?>** încadrează codul sursă al scriptului și sunt obligatorii pentru orice secvență de instrucțiuni PHP. Dacă nu încadrați instrucțiunile PHP între aceste marcaje, serverul de web nu le va interpreta (va trimite spre browser doar textul instrucțiunilor, nu efectul rulării lor).
Funcția echo() e similară cu printf(), dar mai ușor de folosit.

```
<?php
  printf("Hello world!");
?>
```

Pentru a vedea efectul rulării scriptului, introduceți calea și numele lui în spațiul de adresă al browserului de Web. De exemplu: localhost/prog1.php

Variabile

În PHP toate variabilele încep cu simbolul $. Exemple: $nume, $telefon etc.
Numele unei variabile poate începe doar cu o literă sau cu _. Exemple: $a, $_numar etc.
Numele unei variabile poate conține doar litere, cifre si caracterul subliniere _.
Ca și C-ul, PHP este case-sensitive la numele variabilelor (face diferența intre litere mari și mici).

Exemplu de script care calculează media aritmetică a două variabile $a și $b:

```
<?php
    $a=2; $b=3; $media = ($a+$b)/2;
    echo $media;
    //sau mai elegant:
    echo "<br> Media dintre $a si $b este $media  <br>";
?>
```

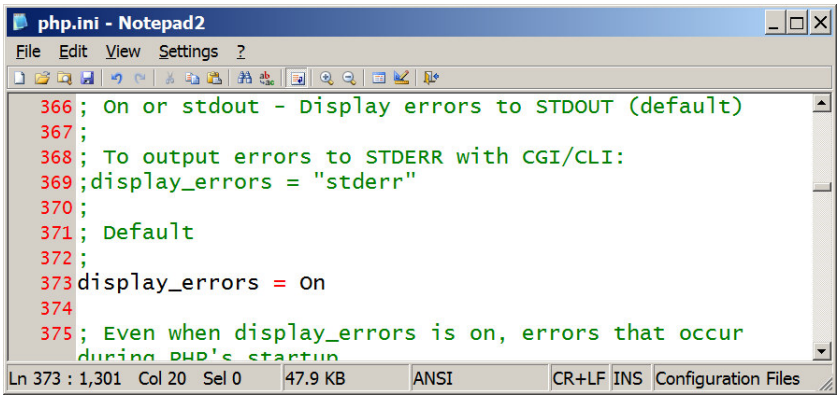
Notă: observați utilizarea marcajelor HTML în funcția *echo*, precum și inserarea directă a variabilelor în textul de afișat. Nu putem folosi marcaje HTML în codul sursă PHP (adică între

<?php și ?>. Dacă dorim folosirea marcajelor HTML direct în fișierul PHP, trebuie să le scriem în afară secvențelor de instrucțiuni PHP, adică sau înaintea <?php sau după ?>. Putem sparge un program PHP în mai multe secvențe (blocuri de instrucțiuni).

Puteți folosi de asemenea funcția C *printf* în loc de echo, de exemplu:

```
printf( "<br> Media dintre %d si %d este %f  <br>",$a, $b, $media);
```

Atenție: în mod predefinit, erorile din program nu vor fi afișate. *Pentru a putea descoperi eventualele erori de sintaxă*, editați fișierul php.ini (din folderul PHP) și setați directiva *display_errors* la *On*, așa cum este arătat în figura următoare. Apoi salvați fișierul php.ini. Nu modificați altceva în php.ini.



Constante

Spre deosebire de variabile, cărora le putem schimba valoarea (chiar și tipul de date) pe parcursul rulării programului, constantele sunt precum sugerează și numele, nemodificabile.

Declararea unei constante se face utilizând funcția *define*(”*nume_ct*”, *valoare*), de exemplu:

```
define("PI", 3.14);
```

După cum se poate observa, în cazul constantelor nu mai este necesară utilizarea simbolului $. De fapt, PHP-ul permite să declarăm o variabilă cu același nume: $PI. Dar această variabilă va fi total independentă de constanta PI definită. De asemenea, se obișnuiește scrierea cu majuscule a constantelor pentru a fi mai ușor identificate în program.

Tipuri de date

În PHP variabilele nu trebuie declarate, deoarece serverul se comportă ca un interpretor (tipul de date este stabilit la atribuirea unei valori).

Principalele tipuri de date sunt:

- Boolean – valoare logică
  $var = True;
  Doar 0 reprezintă False, orice altă valoare numerică este considerată True.
- Integer – număr întreg
  $numar = 20; $b = -10;
- Float – număr real
  $nr_real=2.3; $nr = .3; $c = 1.2e3;
- String – șir de caractere
  $nume = ”Popescu”;
- Array – vector de elemente
- Object – obiect

Putem avea acces la un anume caracter dintr-un șir precizând poziția între [], de exemplu pentru a afișa prima literă din șirul nume: $nume[0], sau a doua literă: $nume[1], sau ultima literă: nume[strlen($nume)-1]

Concatenarea șirurilor de caractere se face cu **.** (punct). Exemplu:

$prenume = ”Ion”; $nume_familie = ”Popescu”; $nume_complet = $prenume**.**” ”**.**$nume_familie;

Vectori

În PHP vectorii pot conține numere sau șiruri de caractere. Modalitatea de definire a vectorilor în PHP este mult mai ușoară și intuitivă față de C. În acest capitol vom păstra însă din PHP doar tipurile de vectori care se definesc identic cu cei din limbajul C.

Exemple:

```
$sir[0]='a';  $sir[1]='b';  $sir[2]='c';
$v[0]=2;  $v[1]=4;  $v[2]=6;
echo  $v[1];  //va afisa al doilea element al vectorului numeric
```

Notă

- Pentru a afișa tot vectorul se poate folosi funcția PHP *print_r()*, de exemplu: *print_r($vector)*.
- La adăugarea unui nou element la un vector, se poate folosi și un indice vid, de exemplu:
  $v[]=8;

Operatori și funcții uzuale

Operatorii clasici din C se pot folosi și in PHP: +, -, *, /, % (restul împărțirii), ++, --.
Operatorii ++ și -- au același efect ca în C dacă sunt amplasați înaintea sau după variabilă.
Câteva funcții uzuale sunt:

- abs($var) // valoarea absolută
- ceil($var) // rotunjire la întregul superior
- floor($var) // rotunjire la întregul inferior
- max($a, $b, $c, …) //maxim
- min($a, $b, $c, …) //minim
- sort($vector) // va returna un vector ordonat crescător
- rsort($vector) // va returna un vector ordonat descrescător
- count($vector) // obținem numărul de elemente din vector

Exerciții

a) Realizați un script PHP care calculează aria unui dreptunghi de lungime și lățime date.
b) Realizați un script PHP care calculează și afișează aria și circumferința cercului de rază R.
Notă: Numărul PI poate fi declarat ca o constantă, o variabilă, sau se poate utiliza funcția pi()

2) Instrucțiuni condiționale și repetitive

Instrucțiunile condiționale din PHP sunt in general identice ca sintaxă cu cele din C, de exemplu:

```
if ($a >= 0)
    echo "$a este pozitiv sau zero <br> ";
else
    echo "$a este negativ <br> ";
```

Dacă trebuie să grupăm mai multe instrucțiuni, putem folosi acoladele { …. }.

Și sintaxa instrucțiunii switch este identică cu C-ul, de exemplu:

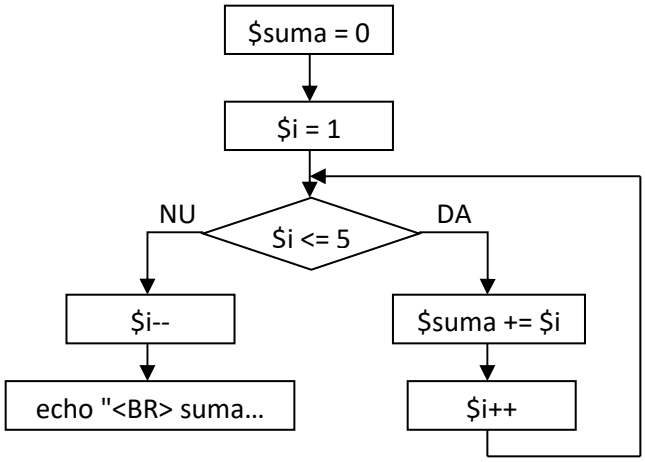
```
switch ($categorie) {
  case 1:
      echo "prima categorie";   break;
  case 2:
      echo "a doua categorie";   break;
  default:
    echo "alta categorie";
}
```

Putem folosi și operatorul condițional, de exemplu: $max =($a > $b)? $a : $b;

Exemple de utilizare a instrucțiunilor repetitive:

```
$suma=0;
for($i=1; $i <= 5; $i++)
    $suma += $i;
$i--;
echo "<BR> suma numerelor de la 1 la $i este $suma <BR>";
```

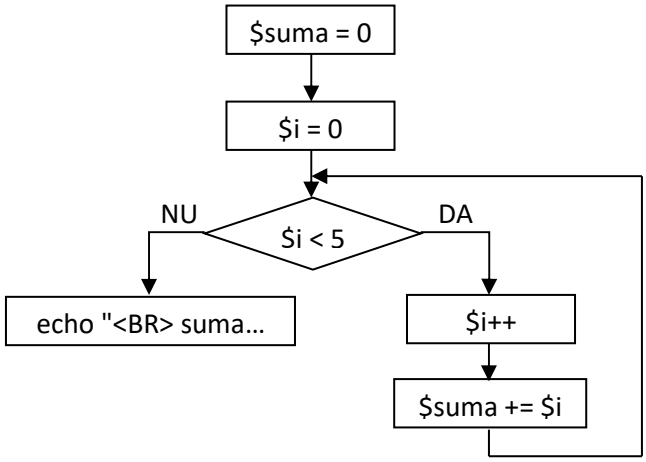
În figura următoare este prezentată schema logică a acestui mic program de calcul al sumei numerelor de la 1 la 5.



Instrucțiunea $i-- care se execută după ieșirea din bucla FOR permite afișarea corectă a propoziției ”suma numerelor de la 1 la 5 este 15”. Dacă omiteam decrementarea variabile $i folosite ca și contor în FOR, propoziția ar fi fost incorectă.

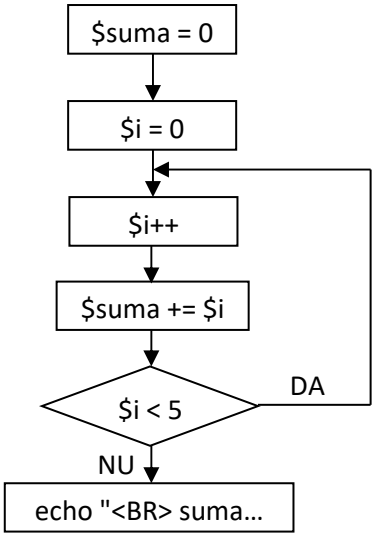
Același efect îl putem obține folosind o buclă de tip *while*:

```
$suma=0; $i=0;
while($i < 5) { $i++;  $suma += $i; }
```



Sau de tip *do-while*:

```
$suma=0; $i=0;
do{
  $i++;   $suma += $i;
}while($i < 5);
```



<u>Exercițiu:</u> realizați un script care afișează elementele unui vector de numere întregi, apoi suma lor.

3) Funcții

<u>Atenție</u>: *definirea funcțiilor în PHP diferă de cea din C sau C++.*

Sintaxa pentru definirea în PHP a unei funcții este următoarea:

```
function  nume_functie(lista variabile)
{
  // instructiuni
}
```

Exemplu:

```
function arie_dreptunghi($l, $L) {
   return $l * $L;
}
```

Începând cu PHP4 funcția poate fi amplasată oriunde în fișier, cu condiția să fie încadrată în marcajele <?php …. ?>
Apelul funcției poate fi, de exemplu: $arie = Arie_Dreptunghi(3,4);
Observați că PHP-ul nu face diferența între litere mari sau mici la numele de funcții.

Variabile locale și globale

O variabilă folosită în interiorul unei funcții este locală acelei funcții, adică nu este vizibilă în exteriorul ei. De asemenea, o variabila locală există doar pe durata execuției funcției. Argumentele unei funcții sunt variabile locale.
Variabilele folosite în exteriorul funcțiilor sunt variabile globale. Dar, aceste variabile nu sunt vizibile în interiorul unei funcții decât dacă se declară „global”.

Exemplu:

```
$x=1;  //variabila globala

function ne_global(){
  echo "<br> x=$x nu e variabila globala <br>";
}

function var_globala(){
  global $x;
  echo "<br> x=$x este o variabila globala <br>";
}

ne_global();   //variabila $x nu va fi gasita
var_globala(); //va fi afisata valoarea variabilei globale $x
```

Există o posibilitate de a păstra valoarea unei variabile locale de la un apel la altul al funcției, prin declararea variabilei de tip „static”.

Exemplu:

```
function var_statica(){
  static $x;
  $x = $x+1;
  echo "<br> x=$x este o variabila locala statica <br>";
}

var_statica(); var_statica(); var_statica();  //apelarea de 3 ori a functiei
```

4) Includerea fișierelor externe în program

PHP-ul permite inserarea automată a conținutului unui fișier ce conține cod sursă PHP / HTML în program. Acest lucru se face prin instrucțiunea *require*(*nume_fisier*). Exemplu:

```
require(”fisier.inc”);
```

Comanda va determina inserarea conținutului fișierului ”fișier.inc” în programul nostru.
Acest fișier poate conține orice fel de cod HTML și/sau PHP.
Mecanismul poate fi util atunci când avem pagini cu un conținut identic (de exemplu antetul sau subsolul paginii); eventuale modificări ale antetului se vor face doar în fișierul inclus, și vor avea

efect în toate paginile unde utilizăm instrucțiunea *require*. De asemenea, în acest mod putem include programe sau funcții scrise de alte persoane, sau funcții de care avem nevoie în mai multe pagini web.

Probleme propuse

1) Realizați o funcție care să returneze maximul din trei numere întregi primite ca parametri ai funcției.

2) Scrieți o funcție care, primind ca argument un caracter, și existând în program un vector (variabilă globală) de caractere, să returneze poziția din vector pe care apare acel caracter, sau -1 dacă nu a fost găsit.

3) Realizați o funcție care primind ca argumente: un vector de caractere ordonat alfabetic, și o variabilă conținând un nou caracter, să returneze vectorul cu noul caracter intercalat pe poziția corespunzătoare (pentru a păstra ordinea alfabetică).

4) Scrieți antetul paginii, precum și codul sursă al unei funcții ce afișează anotimpul curent (primăvară, vară etc.) într-un fișier extern (de exemplu: ”antet.inc”. Includeți apoi acest fișier în programul principal și apelați funcția definită. Exemplu de afișare a lunii curente: echo "
 luna = ".date('m');