**PHP ACADEMY**

**CURS PHP**

**METODE DE PROIECTAREA ALGORITIMILOR PROGRAMAREA STRUCTURATĂ**

**STUDENT: Moldovai Gianina**

**ORADEA**

**2024**

1. **METODE DE PROIECTAREA ALGORITIMILOR – PROGRAMAREA STRUCTURATĂ**

**OPERATORI DE ATRIBUIRE**

Un operator de atribuire în PHP este un operator utilizat pentru a asigna o valoare unei variabile. Cel mai comun operator de atribuire în PHP este operatorul "=".

Operatorul de atribuire "=" atribuie valoarea din partea dreaptă a operatorului variabilei din partea stângă. Sintaxa este următoarea:

De exemplu, pentru a atribui valoarea 5 unei variabile numită "x", folosim:

X= 5;

După asignare, variabila "x" va avea valoarea 5. Operatorul "=" poate fi folosit pentru a atribui orice valoare unei variabile, inclusiv valori întregi, zecimale, șiruri de caractere, valori booleane și chiar și rezultatele unor expresii matematice sau logice.

Este important să reținem că operatorul "=" este un operator de atribuire și nu trebuie confundat cu operatorul de egalitate "==" sau "===" care este utilizat pentru a verifica egalitatea între două valori.

În PHP, există și alți operatori de atribuire, cum ar fi "+=", "-=", "\*=", "/=" și "%=". Acești operatori sunt folosiți pentru a actualiza valoarea unei variabile cu valoarea unei expresii aritmetice. De exemplu, pentru a adăuga valoarea 2 la variabila "x", putem folosi operatorul "+=" în felul următor:

$x +=2; este echivalent cu scrierea $x = $x + 2;

Alte exemple de operatori de atribuire în PHP includ:

- Operatorul "+=": Asignează valoarea rezultatului adunării dintre valoarea curentă a variabilei și o altă valoare. De exemplu: `$x += 2;` echivalează cu `$x = $x + 2;`

- Operatorul "-=": Asignează valoarea rezultatului scăderii dintre valoarea curentă a variabilei și o altă valoare. De exemplu: `$x -= 3;` echivalează cu `$x = $x - 3;`

- Operatorul "\*=": Asignează valoarea rezultatului înmulțirii dintre valoarea curentă a variabilei și o altă valoare. De exemplu: `$x \*= 4;` echivalează cu `$x = $x \* 4;`

Pentru a interschimba valorile a două variabile, putem folosi o variabilă temporară pentru a stoca una dintre valorile și apoi să le schimbăm între ele. De exemplu, pentru a interschimba conținutul a două variabile "x" și "y", putem folosi următorul cod:

$x = 5;

$y = 10;

$temp = $x;

$x = $y;

$y = $temp;

echo "x: " . $x . "<br>"; echo "y: " . $y . "<br>";

În acest exemplu, am creat două variabile "x" și "y" cu valorile inițiale de 5, respectiv 10. Am creat, de asemenea, o variabilă temporară "temp" pentru a stoca valoarea "x". Apoi, am atribuit valoarea "y" la "x" și valoarea "temp" (care conține valoarea inițială a "x") la "y".

La sfârșit, am afișat conținutul actualizat al celor două variabile. Aceasta este o modalitate comună de a interschimba conținutul a două variabile în PHP.

**INSTRUCȚIUNEA IF**

Instructiunea "if" este folosita in programare pentru a executa anumite blocuri de cod doar daca o anumita conditie este indeplinita. In limbajul PHP, sintaxa instructiunii "if" este urmatoarea:

if (conditie) { // blocul de cod care se executa daca conditia este adevarata }

else { // blocul de cod care se executa daca conditia este falsa }

Program pentru calcularea unui triunghi:

$base = 10; // baza triunghiului

$height = 5; // inaltimea triunghiului

$area = 0.5 \* $base \* $height;

if ($area > 0) { echo "Aria triunghiului este: " . $area; }

else {

echo "Datele introduse nu sunt valide pentru a calcula aria triunghiului."; }

Program pentru rezolvarea ecuației de gradul 1:

$a = 2; // coeficientul lui x

$b = 5; // termenul liber

if ($a != 0) {

$x = -($b / $a);

echo "Solutia ecuatiei este: " . $x; }

else {

if ($b == 0) {

echo "Ecuatia are o infinitate de solutii."; }

else { echo "Ecuatia nu are solutie."; } }

Program pentru verificarea daca un numar este par sau impar:

$numar = 7;

if ($numar % 2 == 0) {

echo $numar . " este un numar par."; }

else { echo $numar . " este un numar impar."; }

Program pentru gasirea maximului dintre 3 numere:

$a = 10;

$b = 7;

$c = 12;

$maxim = $a;

if ($b > $maxim) {

$maxim = $b; }

if ($c > $maxim) {

$maxim = $c; }

echo "Maximul dintre " . $a . ", " . $b . " si " . $c . " este: " . $maxim;

**STRUCTURI REPETITIVE**

O structură repetitivă (sau buclă) este o construcție utilizată în programare pentru a executa un bloc de cod de mai multe ori, în funcție de o anumită condiție. Această condiție este verificată înainte de fiecare iterație a buclei, iar blocul de cod este executat atâta timp cât condiția este adevărată.

Structurile repetitive sunt utilizate în mod obișnuit pentru a itera printr-o secvență de valori și pentru a executa un bloc de cod în mod repetat în funcție de o condiție. Acestea sunt utile atunci când trebuie să se efectueze aceeași operație de mai multe ori, cum ar fi sortarea unei liste de elemente sau prelucrarea unui fișier de intrare linie cu linie.

Cele mai comune structuri repetitive utilizate în programare sunt "for", "while" și "do-while". Instrucțiunea "for" este folosită atunci când numărul de iterații este cunoscut înainte de începerea buclei, instrucțiunea "while" este folosită atunci când numărul de iterații nu este cunoscut în avans și instrucțiunea "do-while" este folosită atunci când trebuie să se execute cel puțin o dată blocul de cod din buclă.

1. Bucla "for":

Bucla "for" este o structură repetitivă care permite executarea unui bloc de cod de un număr specific de ori. Sintaxa buclei "for" este următoarea:

for (initializare; condiție; incrementare/decrementare) { // blocul de cod care se execută în fiecare iterație }

Exemplu de utilizare a buclei "for" în PHP pentru afișarea numerelor de la 1 la 5:

for ($i = 1; $i <= 5; $i++) {

echo $i . "\n";

}

1. Bucla "while":

Bucla "while" este o structură repetitivă care permite executarea unui bloc de cod atâta timp cât o anumită condiție este adevărată. Sintaxa buclei "while" este următoarea:

while (condiție) {

// blocul de cod care se execută atâta timp cât condiția este adevărată }

Exemplu de utilizare a buclei "while" în PHP pentru afișarea numerelor de la 1 la 5:

$i = 1;

while ($i <= 5) {

echo $i . "\n";

$i++;

}

**NUMERE NATURALE ÎN PHP**

Primele numere naturale

1. *Afișarea primelor n numere naturale în PHP:*

$n = 10;

for ($i = 1; $i <= $n; $i++) {

echo $i . " "; }

*2. Afisarea primelor n numere pare/impare în PHP:*

a) Pentru numere pare:

$n = 10;

$count = 0;

$num = 2;

while ($count < $n) {

if ($num % 2 == 0) {

echo $num . " ";

$count++; }

$num++; }

b) Pentru numere impare:

$n = 10;

$count = 0;

$num = 1;

while ($count < $n) {

if ($num % 2 != 0) {

echo $num . " ";

$count++; }

$num++; }

1. *Afișarea divizorilor unui număr în PHP:*

$num = 24;

for ($i = 1; $i <= $num; $i++) {

if ($num % $i == 0) {

echo $i . " "; } }

1. *Afișarea numerelor prime până la n în PHP:*

$n = 20;

for ($i = 2; $i <= $n; $i++) {

$isPrime = true;

for ($j = 2; $j <= sqrt($i); $j++) {

if ($i % $j == 0) {

$isPrime = false;

break; } }

if ($isPrime) {

echo $i . " "; } }

1. *Probleme cu cifrele unui număr în PHP: a) Afișarea sumei cifrelor unui număr:*

$num = 12345;

$sum = 0;

while ($num != 0) {

$digit = $num % 10;

$sum += $digit;

$num = floor($num / 10); }

echo "Suma cifrelor este: " . $sum;

b) Afișarea inversului unui număr:

$num = 12345;

$reverse = 0;

while ($num != 0) {

$digit = $num % 10;

$reverse = $reverse \* 10 + $digit;

$num = floor($num / 10); }

echo "Numarul invers este: " . $reverse;