**ლექცია 9 პირობითი და ციკლოს ოპერატორები**

**პირობითი ოპერატორები**

პირობითი ოპერატორები გამოიყენებს სხვადასხვა ქმედებების, ოპერაციების შესას­რულებლათ სხვადასხვა პირობების გათვალისწინებით. ხშირად კოდის შექმნისას დამ­პროგრმებელს სურს სხვადასხვა ქმედებების შესრულება სხვადასხვა გადაწყვეტილებების პირობებში. ასეთ სიტუაციაში, მიზნის მისაღწევად მას შეუძლია გამოიყენოს შემდეგი პირობითი ოპერატორები:

* **if** ოპერატორი - შეასრულებს გარკვეულ კოდს თუ კი სრულდება მითითებული პირობა - ანუ მითითებული პირობა ჭეშმარიტია
* **if...else** ოპერატორი - შეასრულებს გარკვეულ კოდს თუ კი მითითებული პირობა ჭეშმარიტია და მეორე კოდს იმ შემთხვევაში, თუ პირობა მცდარია
* **if...elseif....else** ოპერატორი - ირჩევს შესასრულებლად კოდის ერთ-ერთ ბლოკს რამდენიმე სხვადსხვა ბლოკისგან
* **switch** ოპერატორი - შესასრულებლად ირჩევს კოდის რამდენიმე ბლოკიდან ერთ-ერთს

**if ოპერატორი**

**if** ოპერატორი გამოიყენება გარკვეული კოდის შესასრულებლად, მხოლოდ იმ შემთხვევაში, თუ კი მითითებული პირობა ჭეშმარიტია.

სინტაქსისი

if (პირობა) { *კოდი, რომელიც შესრულდება თუ კი პირობა ჭეშმარიტია ;*  
}

ფიგურული ფრჩხილები კოდის ბლოკს აღნიშნავს და ძირითადად მაშინ გამოიყენება , როცა ბლოკში შესასრულებელია რამდენიმე ოპერაცია. ამდენად, თუ კი პირობის გათვა­ლისწინებით საჭიროა მხოლოდ ერთი ოპერაციის შესრულება, მაშინ ფიგურული ფრჩხი­ლების გამოყენება სავალდებულო არაა.

ვთქვათ ავტონაწილების მაღაზიაში შესრულებული შეკვეთის შესახებ კლიენტს უბრუნ­დება ინფორმაცია მისი შეკვეთის შესახებ. ჯერ ერთი, ჩავწეროტ მისადმი მისალმება ”დღე მშვიდობისა”, თუ კი ვაგზავნით ამ ინფორმაციას 20:00 საათამდე; მეორეც - უნდა გავით­ვალისწინოთ, ის შემთხვევა, როცა შემკვეთმა შემთხვევით დააჭირა ხელი ”გაგზავნას” და ფაქტიურად არაფერი არ შეუკვეთია. პროგრამული კოდი სკრიპტში იქნება :

ლისტინგი 1. **if** ოპერატორი

<?php  
$t=date("H");  
  
if ($t<"20") {  
   echo "დღე მშვიდობისა!<br>";  
}

if ($totalqty==0)

echo '<font color="red"><h3 align="center">თქვენ არაფერი არ შეგიკვეთიათ წინა გვერდზე!</h3></font><br>';?>

ჩავსვათ ლისტინგი 1 ლექცია 8-ის ლისტინგ 13-ში და ლექცია 8-ის ლისტინგ 12-ის გაშვებისას - <http://localhost/laboratoriuli_9/lab_9_1_if_operator/SekveTa.html> - თუ კი არაფერს არ შევავსებთ - მივიღებთ:

**ავტონაწილები ბუბასგან**

**შეკვეთის შედეგი**

დღე მშვიდობისა!

შეკვეთა დამუშავდა10:10, 9th November 2014

შეკვეთილია:

0 საბურავი  
0 ბოთლი ზეთი  
0 აალების სანთელი

**თქვენ არაფერი არ შეგიკვეთიათ წინა გვერდზე!**

შეკვეთილი საქონელი: 0  
ჯამში: $0.00  
მთლიანად, გაყიდვებისგან გადასახადის ჩათვლით: $0.00

შედეგი 1. if ოპერატორის გამოყენება

**ოპერატორი else**

თუ კი if ოპერატორის საშუალებით ჩვენ შეგვიძლია განვსაზღვროთ, უნდა შეს­რულდეს ესა თუ ის ოპერაცია პირობისდა მიხედვით, else ოპერატორის საშუალებით შესაძლებელია პირობისდა მიღედვით ავირჩიოთ ესა თუ ის განვითარება სცენარისა - კოდის ბლოკების შესრულება. else ოპერატორის სინტაქსისია:

**if** (პირობა) {  
  შესრულდება კოდი, თუ კი პირობა ჭეშმარიტია;  
}

**else** {  
  შესრულდება კოდი, თუ კი პირობა მცდარია;

}

ყოველი if -ის შემდეგ შესაძლებელია გამოყენებულ იქნეს მხოლოდ ერთი else. მოვახდინოთ ლისტინგ 1-ის მოდიფიცირება - თუ შეკვეთა დამუშავდა საღამოს - 20:00 საათის შემდეგ, გამოვიტანოთ შეტყობინება - ”საღამო მშვიდობისა”, ამასთან თუ კი არავითარი შეკვეთა არაა, არ გამოვიტანოთ შესაბამისი შეკვეთების რაოდენობა, დარჩეს მხოლოდ შეტყობინება - "თქვენ არაფერი არ შეგიკვეთიათ წინა გვერდზე!”, ხოლო შეკვეთის რაოდენობა გამოვიდეს იმ შემთხვევაში, თუ კი შეკვეთა რეალურად დაფიქსირდა:

ლისტინგი 2. else ოპერატორი

<?php  
if ($t<"20") {

echo "დღე მშვიდობისა!<br>";

}

else{

echo "საღამო მშვიდობისა";

}

if ($totalqty==0)

{

echo '<font color="red"><h3 align="center">თქვენ არაფერი არ შეგიკვეთიათ წინა გვერდზე!</h3></font><br>';

}

else

{

echo '<br><p>შეკვეთილია: </p>';

echo $tireqty.' საბურავი<br />';

echo $oilqty.' ბოთლი ზეთი<br />';

echo $sparkqty.' აალების სანთელი<br />';

echo '<br>';

}

?>

ლისტინგ 2-ს ჩავსვამთ ლისტინგ 12-ში და შეკვეთის ფორმის გაშვების შემდეგ -http://localhost/laboratoriuli\_9/lab\_9\_2\_else\_operator/SekveTa.html, მას მერე რაც შევავსებთ ფორმის ველებს 4 საბურავი: 5 აალების სანთელი - მივიღებთ

# ავტონაწილები ბუბასგან

**შეკვეთის შედეგი**

დღე მშვიდობისა!

შეკვეთა დამუშავდა10:57, 9th November 2014

შეკვეთილია:

4 საბურავი  
ბოთლი ზეთი  
5 აალების სანთელი  
  
შეკვეთილი საქონელი: 9  
ჯამში: $420.00  
მთლიანად, გაყიდვებისგან გადასახადის ჩათვლით: $462.00

შედეგი 2. else ოპერატორის გამოყენება

როგორც ვხედავთ, გამოვიდა შეკვეთილი საქონელის დასახელევა და მათი რა­ოდენობა. იმ საქონელის დასახელებაც კი, რომელიც საერთოდ არ შეგვიკვეთია. დავსვათ კითხვა, შესძლებელია კი ის შეკვეთა არ გამოვიტანოთ, რომელიც არ გან­გვიხორციელებია? ანუ შესაბამისი საქონელიმ არ მოგვითხოვია. პასუხი დადებითია და ამ შემთხვევაში კოდის ბლოკებში შეგვიძლია კვლავ გამოვიყენოთ else ან if ოპერატორი. ვანახოთ მისი გამოყენების მაგალითი:

ლისტინგი 3. if else ოპერატორის ჯაჭვი

<?php  
if ($t<"20") {

echo "დღე მშვიდობისა!<br>";

}

else{

echo "საღამო მშვიდობისა";

}

if ($totalqty==0)

{

echo '<font color="red"><h3 align="center">თქვენ არაფერი არ შეგიკვეთიათ წინა გვერდზე!</h3></font><br>';

}

else

{

echo '<br><p>შეკვეთილია: </p>';

if ($tireqty>0)

echo $tireqty.' საბურავი<br />';

if ($oilqty >0)

echo $oilqty.' ბოთლი ზეთი<br />';

if ($sparkqty >0)

echo $sparkqty.' აალების სანთელი<br />';

echo '<br>';

}

?>

მოვახდინოთ ლექცია 8-ში ლისტინგ 13-ში ლისტინგ 3-ის ჩაწერა და გავუშვებთ რა შეკვეთის ფორმას localhost/laboratoriuli\_9/lab\_9\_3\_if\_else\_jaWvi/SekveTa.html მონაცემებით საბურავი 4: ზეთი - ცარიელი: აალების სანთლები - 5, მივიღებთ:

**ავტონაწილები ბუბასგან**

**შეკვეთის შედეგი**

დღე მშვიდობისა!

შეკვეთა დამუშავდა11:17, 9th November 2014

შეკვეთილია:

4 საბურავი  
5 აალების სანთელი  
  
  
შეკვეთილი საქონელი: 9  
ჯამში: $420.00  
მთლიანად, გაყიდვებისგან გადასახადის ჩათვლით: $462.00

შედეგი 3. If else ოპერატორის ჯაჭვი

როგორც ვხედავთ ან გამოჩნდა წარწერა - ”ზეთი”, რადგანაც ის შეკვეთილი არ გვქონდა.

## if...elseif....else ოპერატორი

## როგორც ვიცით, ყოველი if -ის შემდეგ შესაძლებელია გამოყენებულ იქნეს მხოლოდ ერთი else. მაგრამ თუ კი საჭიროა რამდენიმე (ორზე მეტი) ვარიატის არჩევა პირობისდა შესაბამისად, მაშინ გამოიყენება ოპერატორი, რომლის სინტაქსისია:

## if (პირობა) {   შესრულდება კოდი, თუ პირობა არის *true;* } elseif (პირობა1) {   შესრულდება კოდი, თუ პირობა არის *true;* }

## [elseif (პირობა2) {   შესრულდება კოდი, თუ პირობა არის *true;*

## }]

## ........

## [elseif (პირობაn) {   შესრულდება კოდი, თუ პირობა არის *true;*

## }]

## [else {   შესრულდება კოდი, თუ პირობა if -თან არის *false;* }]

შევნიშნოთ, რომ ამ კონსტრუქციაში **else** შეიძლება არც კი იყოს გამოყენებული. მოვიჱვანოთ მაგალითები, ისევ ავტონაწილების მაღაზიაში შეკვეთის გათვალისწინებით. თუ შეკვეთის შესახებ ინფორმაცია გაიცემა დღის 10:00 საათამდე, მივესალმოთ შემდეგნაირად - ”დილა მშვიდობისა!”, ხოლო იმისდა მიხედვით, თუ რა რაოდენობის საბურავები იქნა შეკვეთილი, გავითვალისწინოთ შეღავათი- დისკაუნტი:

* შეკვეთილია 10-ზე ნაკლები საბურავი - შეღავათი არაა;
* შეკვეთილია 10-დან 49 საბურავი - შეღავათი 5%;
* შეკვეთილია 50-დან 99 საბურავი - შეღავათი 10%;
* შეკვეთილია 100 მეტი საბურავი - შეღავათი 15%;

შევადგინოთ კოდი რომელიც ითვალისწინებს ამ მოთხოვნებს, ამასთან შევადგინოთ, რთული ლოგიკური გამოსახულება და კავშირით (&& და and-ის საშუალებით)

ლისტინგი 4. **elseif** ოპერატორი

<?php

$t=date("H");

if ($t<"10") {

echo "დილა მშვიდობისა";

} elseif ($t<"20"){

echo "დღე მშვიდობისა!<br>";

} else{

echo "საღამო მშვიდობისა";

}

$discount=0;

if($tireqty<10){

echo"შეღავათი საბურავებზე არ გეკუთვნით";

}elseif(($tireqty>=10)&&($tireqty<=49)){

$discount=5;echo 'გეკუთვნით შეღავათი საბურავებზე '."$discount".' %<br>';

}elseif(($tireqty>=50)and($tireqty<=99)){

$discount=10; echo 'გეკუთვნით შეღავათი საბურავებზე '."$discount".' %<br>';

}elseif($tireqty>100){

$discount=15;echo 'გეკუთვნით შეღავათი საბურავებზე '."$discount".' %<br>';

}

?>

ეს კოდი ჩასმულია ლისტინგ 13-ისა ლექცია 8-ში და ვღებულობთ

ლისტინგ 4’ -ს

<html>

<head>

<title>ავტონაწილები ბუბასგან — შეკვეთის შედეგები</title>

</head>

<?php

// ცვლადების მოკლე სახელების შექმნა

$tireqty = $\_POST['tireqty'];

$oilqty = $\_POST['oilqty'];

$sparkqty = $\_POST['sparkqty'];

?>

<body>

<h1>ავტონაწილები ბუბასგან</h1>

<h2>შეკვეთის შედეგი</h2>

<?php

$t=date("H");

if ($t<"10") {

echo "დილა მშვიდობისა";

} elseif ($t<"20"){

echo "დღე მშვიდობისა!<br>";

} else{

echo "საღამო მშვიდობისა";

}

echo "<p>შეკვეთა დამუშავდა".date('H:i, jS F Y')."</p>";

$totalqty = 0;

$totalqty = $tireqty + $oilqty + $sparkqty;

if ($totalqty==0)

{

echo '<font color="red"><h3 align="center">თქვენ არაფერი არ შეგიკვეთიათ წინა გვერდზე!</h3></font><br>';

}

else

{

echo '<br><p>შეკვეთილია: </p>';

if ($tireqty>0)

echo $tireqty.' საბურავი<br />';

if ($oilqty >0)

echo $oilqty.' ბოთლი ზეთი<br />';

if ($sparkqty >0)

echo $sparkqty.' აალების სანთელი<br />';

echo '<br>';

echo '<br>';

}

$discount=0;

if($tireqty<10){

echo"შეღავათი საბურავებზე არ გეკუთვნით";

}elseif(($tireqty>=10)&&($tireqty<=49)){

$discount=5;echo 'გეკუთვნით შეღავათი საბურავებზე '."$discount".' %<br>';

}elseif(($tireqty>=50)and($tireqty<=99)){

$discount=10; echo 'გეკუთვნით შეღავათი საბურავებზე '."$discount".' %<br>';

}elseif($tireqty>100){

$discount=15;echo 'გეკუთვნით შეღავათი საბურავებზე '."$discount".' %<br>';

}

echo "შეკვეთილი საქონელი: ".$totalqty."<br>";

$totalamount = 0.00;

define('TIREPRICE', 100);

define('OILPRICE', 10);

define('SPARKPRICE', 4);

$totalamount = $tireqty \* TIREPRICE\*(1-$discount/100)

+ $oilqty \* OILPRICE

+ $sparkqty \* SPARKPRICE;

echo "ჯამში: $".number\_format($totalamount,2)."<br />";

$taxrate = 0.10; // ადგილობრივი მოსაკრებელი გაყიდვებისგან შეადგენს 10%

$totalamount = $totalamount \* (1 + $taxrate);

echo "მთლიანად, გაყიდვებისგან გადასახადის ჩათვლით: $".number\_format($totalamount,2)."<br />";

?>

</body>

</html>

ამ PHP კოდით დამუშავკება შეკვეთის ფორნა - როცა საბურავი არის 60 და აალების სანთელი 6. შედეგად მივიღებთ:

# ავტონაწილები ბუბასგან

**შეკვეთის შედეგი**

დღე მშვიდობისა!

შეკვეთა დამუშავდა12:20, 9th November 2014

შეკვეთილია:

60 საბურავი  
6 აალების სანთელი  
გეკუთვნით შეღავათი საბურავებზე 10 %  
შეკვეთილი საქონელი: 66  
ჯამში: $5,424.00  
მთლიანად, გაყიდვებისგან გადასახადის ჩათვლით: $5,966.40

შედეგი 4’. elseif ოპერატორის გამოყენება

elseif ოპერატორის კასკადური ერთობლიობის გამოყენებისას, პრინციპში, უნდა სრულდებოდეს კოდის ერთ-ერთი ბლოკი ან ოპერატორი, რადგანაც პირობები უნდა იყვნენ ურთიერთგამომრიცხავი და მხოლოდ ერთი პირობა უნდა იყოს ჭეშმარიტი. მაგრამ თუ კი აღმოჩდა, რომ ერთდროულად რამდენიმე პირობა იღებს მნიშვნელობას **true**, მაშინ შესრულდება მხოლოდ ის ბლოკი ან ოპერატორი, რომელიც პირველი ასეთი პირო­ბის შემდეგაა.

**პირობითი ტერნატული ოპერატორი ”?”**

PHP-ში if ... else ოპერატორთან ერთად განიხილება ტერნარული ოპერატორი, რომელშიც საკვანძო სიტყვები ”if” და ”else” შესაბამისად შეცვლილია ”**?**” და ”**:**” სიმბოლოებით. ტერნარული ოპერატორის სინტაქსისია:

გამოსახულება 1 ? გამოსახულება2 : გამოსახულება3;

თავდაპირველად გამოითვლება გამოსახულება 1 და თუ კი ის ჭეშმარიტია, მაშინ შესრულდება გამოსახულება2 და მიღებული შედეგი იქნება რეზულტატი , წინააღ­მდეგ შემთხვევაში შესრულდება გამოსახულება3 და მიღებული შედეგი იქნება რეზულტატი.

ტერნარული ოპერატორის გამოყენებით, კოდი:

<?php  
if ($t<"20") {

echo "დღე მშვიდობისა!<br>";

}

else{

echo "საღამო მშვიდობისა";

}

?>

შეიძლება ჩაიწეროს შემდეგნაირად:

echo ($t<"20") ? "დღე მშვიდობისა!<br>": "საღამო მშვიდობისა"; როგორც ვხედავთ, ვღებულობთ უფრო კომპაქტურ ჩანაწერს. მოვიყვანოთ კიდევ ერთი მაგალითი, აბსოლუტური მნიშვნელობის გამოთვლა:

<?php

$value=-2,5;

$abs\_value=Svalue<0? -$value : $value;

echo $abs\_value;

?>

ოპერატორი switch / გადამრთველი ოპერატორი

if ოპერატორის პირობამ შეიძლება მიიღოს ბულიანის მხოლოდ ერთ-ერთი მნიშ­ვნელობა. მაგრამ არაა გამორიცხული, რომ სცენარის გაგრძელება იყოს პირობის რამ­დენიმე, ორზე მეტი, შესაძლო მნიშვნელობებისთვის. ოპერატორი switch-ში, პირობამ შეიძლება მიიღოს რამდენიმე მნიშვნელობა და ამასთან ამ მნიშვნელობის ტიპი უნდა იყოს მარტივი-სკალარული: integer, string და float. პირობის რამდენიმე მნიშვნელობიდან ერთ-ერთის შესრულებისთვის უნდა შევარჩიოთ შესაბამისი case ოპერატორი, და უნდა ჩავწეროთ, მაგრამ სავალდებულო არაა, რადგანაც ჩუმათობის პრინციპით, შესრულდება იმ შემთხვევა­ში, თუ კი პირობის არცეთი მნიშვნელობა არა ემთხვევა case-ს. ოპერატორი switch -ის სინტაქ­სისია:

switch(***გამოსახულება***) // გადამრთველის გამოსახულება

{

case value1: // გამოსახულება 1

statements; // ოპერატორების ბლოკი

break;

case value2: // გამოსახულება 2

statements;

break;

default:

statements;

}

მმართველი სტრუქტურა switch მართვას გადასცემს case -ით მონიშნულ იმ ოპერატოტს ან ოპერატორების ბლოკს, რომლის კონსტანტური ტიპის მნიშვნელობა value ემთხვევა გადამ­რთველის გამოსახულების მნიშვნელობას. თავიდან ხდება ***გამოსახულება*** ანალიზი, და ხდე­ბა გადასვლა იმ შემთხვევაზე - case, რომლის შემდეგ ჩაწერილი კონსტანტური ტიპის მნიშ­ვნელობა value ტოლია გადამრთველის გამოსახულების მნიშვნელობას. იწყება ოპერატორთა შესაბამისი ბლოკის შესრულება, მანამ სანამ არ გამოჩნდება საკვანძო სიტყვა break, რომელიც აღნიშნავს გამოსვლას switch კონსტრუქციიდან. თუ კი გადამრთველის გამოსახულების მნიშვნელობა არ ემთხვევა არც ერთ კონსტანტურ მნიშვნელობას, მაშინ მართვა გადაეცემა ოპერატორთა იმ ბლოკს, რომელიც აღნიშნულია default საკვანძო სიტყვით. ყოველ გადამ­რთველში უნდა იყოს არა უმეტეს ერთი default საკვანძო სიტყვისა. **break** და **default** სავალდებულო საკვანძო სიტყვები არა და ისინი შეიძლება არც იყვნენ გადამრთველში.

განვიხილოთ გადამრთველის გამოყენების მაგალითი ავტონაწილების მაღაზიისთვის: ვთვათ მფლობელ ბუბას სურს იცოდეს, რა გზით აღწევენ მასთან კლიენტენი, ამისთვის გამოვყოთ შეკვეთის ფორმის კოდში შესაბამისი ნაწილი, რომელიც აფიქსირებს იმას, თუ საიდან მონახა მან ბუბას მაღაზია:

<tr>

<td>როგორ გახვედით ბუბაზე?</td>

<td>

<select name="find">

<option value = "a">მე მუდმივი კლიენტი ვარ</option>

<option value = "b">სატელევიზიო რეკლამით</option>

<option value = "c">სატელეფონო ცნობარში</option>

<option value = "d">ზეპირი რეკომენდაცია</option>

</select>

</td>

</tr>

ამდენად გზა, რომელიც ჩაიწერება ბუბას მაღაზიაში მოხვედრისა, მოთავსდება ცვლადში - $find, რომელიც POST მეთოდით გადაეცემა სერვერს და იქ ის ჩაიწერება $\_ POST მასივში. ეს $find ცვლადი შეიძლება დამუშავდეს if და elseif ოპერატორებით, მაგალი­თად:

if ($find=="a") {

echo "<p>მუდმივი კლიენტი.</p>";

} elseif ($find=="b") {

echo "<p>კლიენტი სატელევიზიო რეკლამის შედეგად მოვიდა.</p>";

}elseif ($find=="c") {

echo "<p>კლიენტი სატელელეფონო ცნობარის შედეგად მოვიდა.</p>";

}elseif ($find=="d") {

echo "<p>კლიენტი სხვა პირის რეკომენდაციის შედეგად მოვიდა.</p>";

}else {

echo "<p>გაუგებარია კლიენტი როგორ მოხვდა ჩვენთან.</p>";

}

გადამრთველის საშუალებით კი ის შეიძლება შემდეგნაირად ჩაიწეროს

ლისტინგი 5. switch ოპერატორი

$find=$\_POST['find'];

switch ($find) {

case "a" :

echo "<p>მუდმივი კლიენტი.</p>";

break;

case "b" :

echo "<p>კლიენტი სატელევიზიო რეკლამის შედეგად მოვიდა.</p>";

break;

case "c" :

echo "<p>კლიენტი სატელელეფონო ცნობარის შედეგად მოვიდა.</p>";

break;

case "d" :

echo "<p>კლიენტი სხვა პირის რეკომენდაციის შედეგად მოვიდა.</p>";

break;

default :

echo "<p>გაუგებარია კლიენტი როგორ მოხვდა ჩვენთან.</p>";

break;

}

ლისტინგი 5-ის მოთავსებებისას ფორმის ცვლადების დამუშავების კოდში და შემდეგ ამ შეკვეთის შესრულება <http://localhost/laboratoriuli_9/lab_9_5_switch_operator/SekveTa.html> გვაძლევს შედეგს:

**ავტონაწილები ბუბასგან**

**შეკვეთის შედეგი**

დღე მშვიდობისა!

შეკვეთა დამუშავდა11:19, 10th November 2014

შეკვეთილია:

4 საბურავი  
8 ბოთლი ზეთი  
6 აალების სანთელი  
  
შეღავათი საბურავებზე არ გეკუთვნითშეკვეთილი საქონელი: 18  
ჯამში: $504.00  
მთლიანად, გაყიდვებისგან გადასახადის ჩათვლით: $554.40

კლიენტი სხვა პირის რეკომენდაციის შედეგად მოვიდა.

შედეგი 5. switch ოპერატორის გამოყენება

**ციკლის ოპერატორები**

ციკლის ოპერატორები გამოიყენება განმეორებადი ოპერაციების, ბლოკი ოპერა­ციების ავტომატიზაციისთვის. ხშირად პროგრამისტს, სჭირდება მრავალჯერ შეას­რულოს ოპერაციების ერთი და იგივე მიმდევრობა. ამ დროს გამოიყენება ციკლის ოპერატორები.

PHP-ში არის შემდეგი ტიპის ციკლის ოპერატორები:

* **while** - მეორდება კოდის ბლოკი მანამ სანამ სპეციალური პირობა არის ჭეშმარიტი
* **do...while** - კოდის ბლოკი შესრულდება ერთხელ მაინც და შემდეგ მეორდება მანამ, სანამ სპეციალური პირობა არის ჭეშმარიტი
* **for** - კოდის ბლოკი მეორდება მითითებულ რიცხვჯერ
* **foreach** - კოდის ბლოკკის გამეორება მასივის ყოველი ელემენტისთვის

**ციკლი while**

სინტაქსისი:

while (პირობა სამართლიანის) {  
*შესასრულებელი კოდი*;  
}

ციკლი while გამოყენების მაგალითი მოვიყვანოთ ისევ ავტონაწილების მაღაზიასთან დაკავშირებით: ვთქვქთ მფლობელმა ბუბამ გადაწყვიტა ჰქონდეს ცხრილი, რომელიც ასახავს მიტანის ღირებულებას დამოკიდებულს მანძილზე, რა თქმქ უნდა კომერსანტი ბუბა ამ თანხას კლიენტს გადაახდევინებს! ამდენად გვაქვა დამოკიდებულება შეკვეთილი საქონსლის მიტანის კურიერული საფასური, დამოკიდებული მანზილზე, რომელიც ასე გამოიყურება

ცხრილი 1. მანძილისა და საფასურის კავშირი

|  |  |
| --- | --- |
| მანძილი | საფასური |
| 50 | 5 |
| 100 | 10 |
| 150 | 15 |
| 200 | 20 |
| 250 | 25 |

ცხრილი 1-ის გამოტანის HTML - კოდი.

ლისტინგი 6 მანძილისა და საფასურის გამოტანის HTML კოდი gadazidva.html

**<html>**

**<body>**

**<table border = "0" cellpadding = "3">**

**<tr>**

**<td bgcolor = "#CCCCCC" align = "center">მანძილი</td>**

**<td bgcolor = "#CCCCCC" align = "center">საფასური</td>**

**</tr>**

<tr>

<td align = "right">50</td>

<td align = "right">5</td>

</tr>

<tr>

<td align = "right">100</td>

<td align = "right">10</td>

</tr>

<tr>

<td align = "right">150</td>

<td align = "right">15</td>

</tr>

<tr>

<td align = "right">200</td>

<td align = "right">20</td>

</tr>

<tr>

<td align = "right">250</td>

<td align = "right">25</td>

</tr>

</table>

</body>

</html>

ლისტინგი 6 -ის შესრულების შედეგი

|  |  |
| --- | --- |
| მანძილი | საფასური |
| 50 | 5 |
| 100 | 10 |
| 150 | 15 |
| 200 | 20 |
| 250 | 25 |

შედეგი 6. html – კოდში შედგენილი ცხრილის გამოტანა

მოვახდინოთ ლისტინგ 6-ში ერთი და იგივე ოპერაციების განმეორება ჩავწეროთ

ლისტინგ 7-ის gadazidva.php სახით

<html>

<body>

<table border="0" cellpadding="3">

<tr>

<td bgcolor = "#CCCCCC" align = "center">მანძილი</td>

<td bgcolor = "#CCCCCC" align = "center">საფასური</td>

</tr>

<?php

$distance = 50;

**while** ($distance <= 250) **{**

echo "<tr>

<td align=\"right\">".$distance."</td>

<td align=\"right\">".($distance / 10)."</td>

</tr>\n";

$distance += 50;

**}**

?>

</table>

</body>

</html>

ლისტინგ 7-ის გაშვებით <http://localhost/laboratoriuli_9/lab_9_6_while_operator/gadazidva.php>

მივიღებთ იგივე შედეგს, რაც ლისტინგ 6-ის გაშვებით. ლისტინგ 7-ში \n-ის მოთავსებით ორ­მაგ ბრჭყალებში, მივუთითებთ, რომ საჭიროა ახალ სტრიქონზე გადასვლა. შევნიშნოთ, რომ ”\n” გამოყენება Unix -ში ახალ სტრიქონზე გადასასვლელად. ხოლო Windows-ში და TCP/IP პროტოკოლებში საჭიროა გამოვიყენოთ კომბინაცია “\r\n” , სადაც “\r” კარეტის დაბრუნებას აღნიშნავს. $distance ცვლადი ამ კოდში თავისებურ მთვლელს წარმოადგენს, ის ციკლის გა­რეთ იქმნება თავდაპირველად და შემდეგ ის **ციკლის ტანი**ს - (ფიგურულ ფრჩხილში მოთავ­სებული კოდის ნაწილი) შესრულებისას უნდა შეიცვალოს და მერე კვლავ შემოწმდება პირო­ბა. თუ არ ხდება მთვლელის შეცვლა, მაშინ ციკლის ტანი მუდამ შესრულდება, თუ კი ის თა­ვიდანვე აკმაყოფილებდა პირობას - ამ შემთხვევაში ამბობენ, რომ მოხდა ჩაციკვლა, რაც არასასურველია!

**do...while ციკლი**

do...while ციკლის სინტაქსისია:

do {  
*შესასრულებელი კოდი;*} while (პირობა);

do...while ციკლის შესრულებისას, თავდაპირველად შესრულდება კოდი რომელიც მოთავ­სებულია ფიგურულ ფრჩხილებში, შენდეგ ხდება პირობის შემოწმება, თუ კი პირობა ჭეშ­მარიტია, მაშინ ხდება ისევ do -ს შემდეგ ფიგურულ ფრჩხილებში მოთავსებული კოდის შესრულება. აღსანიშნავია, რომ do -ს შემდეგ ფიგურულ ფრჩხილებში მოთავსებული კოდი ერთხელ მაინც შესრულდება, იმ შემთხვევაშიც კი, რონ თავიდანვე პირობა მცდარია, მოვიყვანოთ მაგალითი:

ლისტინგი 8. do...while ციკლი

<!DOCTYPE html>

<html>

<body>

**<?php**

**$x=7;**

**do {**

**echo "ცვლადი \$x=$x <br>";**

**$x=$x-2;**

**} while (($x<=5)and($x>=1));**

**?>**

</body>

</html>

ლისტინგ 8-ის გაშვებით <http://localhost/laboratoriuli_9/lab_9_7_do_while_operator/do_while.php> ვღებულობთ:

ცვლადი $x=7   
ცვლადი $x=5   
ცვლადი $x=3   
ცვლადი $x=1

შედეგი 8.do…while – კოდის მუშაობის შედეგის გამოტანა

მივაქციოთ ყურადღება გამოტანილ პირველ სტრიქონს, ცვლადის ეს მნიშვნელობა არ ჯდება ციკლის პირობაში, მაგრამ მაინც შესრულდა, რადგანაც პირველად იყო ჩაწერილი - ამ შემთხვევაში კი, ის უცილობლად უნდა შესრულდეს.

**for ციკლი**

for ციკლი გამოიყენება მაშინ, როცა წინასწარაა ცნობილი, თუ რამდენჯერ უნდა შესრულდეს ციკლის ტანი - კოდის ბლოკი.

სინტაქსისი:

**for** (საწყისი მთვლელი*;* მთვლელის ტესტი; მთვლელის მნიშვნელობის ცვლილება) **{**  
  ციკლის ტანი - შესასრულებელი კოდი*;*  
**}**

1. საწყისი მთვლელი: ხდება მთვლელის საწყისი მნიშვნელობის ინიციალიზაცია;
2. მთვლელის ტესტი: შეფასდება ციკლის ყოველი იტერაცია, თუ შეფასებისას მივიღეთ true მაშინ ციკლის განი შესრულდება, თუ კი შეფასებისას მივიღეთ false, მასინ იტერაცია არ შესრულდება და ციკლი დასრულდება;
3. მთვლელის მნიშვნელობის ცვლილება: ხდება მთვლელის მნიშვნელობის ცვლილება და შემდეგ გადავდივართ მეორე პუნქტზე.

ჩვენ შეგვიძლია გადავწეროთ ლისტინგ 7-ში while ციკლი for ციკლით, მივიღებთ

ლისტინგი 9. for ციკლი

<html>

<body>

<table border="0" cellpadding="3">

<tr>

<td bgcolor = "#CCCCCC" align = "center">მანძილი</td>

<td bgcolor = "#CCCCCC" align = "center">საფასური</td>

</tr>

<?php

for($distance = 50;$distance <=250;$distance+=50){

echo "<tr>

<td align=\"right\">".$distance."</td>

<td align=\"right\">".($distance / 10)."</td>

</tr>\n";

}

?>

</table>

</body>

</html>

ლისტინგი 9-ის შესრულებისას <http://localhost/laboratoriuli_9/lab_9_8_for_operator/for_operator.php> მივიღებთ იმაცე შედეგს, რაც ლისტინგ 7-ის გაშვებისას.

**foreach ციკლი**

foreach ციკლი გამოიყენება ხდება მხოლოდ მასივისთვის და ციკლი მუშაობს წყვილით გასაღები/მნიშვნელობა.

სინტაქსისი:

foreach ($*array* as$*value*) {  
  ჩასათვირთი კოდის ბლოკი*;*  
}

ციკლის ყოველი იტერაციისას მასივის მიმდინარე ელემენტს მიენიჭბა მნიშვნელობა $*value* და მასივის მიმთითებელი შეიცვლება და სე მასივის მიმთითებლის ბოლომდე.სემდეგ მაგალითში ხდება მასივისთვის მინიჭებული ელემენტების გამოტანა:

ლისტინგი 9. foreach ციკლი

<!DOCTYPE html>

<html>

<body>

<?php

$colors = array("წითელი","მწვანე","ცისფერი","ყვითელი");

foreach ($colors as $value) {

echo "$value <br>";

}

?>

</body>

</html>

ლისტინგ 9-ის გაშვებით <http://localhost/laboratoriuli_9/lab_9_9_foreach_operator/foreach_operator.php> მივიღებთ:

**წითელი   
მწვანე   
ცისფერი   
ყვითელი**

შედეგი 9. foreach – კოდის მუშაობის შედეგის გამოტანა

**მმართველი სტრუქტურიდან და სცენარიდან გამოსვლა**

თუ კი დადგა საჭიროება შეჩერდეს რომელიმე ფრაგმენტის შესრულება, შესაძლებე­ლია სამიდან ერთ-ერთი საშუალების გამოყენება.

თუ კი საჭიროა ციკლის დასრულება , მაშინ უნდა ჩაისვას სამოსამსახურე სიტყვა - **break**, რომელსაც ჩვენ უკვე შევხვდით ოპერატორ switch -ში. ციკლში break ოპერატორის შესრულებისას, ხდება ციკლიდან გამოსვლა და მართვა გადაეცემა სცენარში ამ ციკლის შემდეგ ჩაწერილ ბრძანებას.

ხოლო თუ კი ციკლიდან გამოსვლა არ არის სასურველი, და საჭიროა, შემდეგი იტერაციის შესრულება, მაშინ უნდა იქნეს გამოყენებული ოპერატორი **continue**.

მთლიანი PHP სცენარის დასრულებისთვის გამოიყენება ოპერატორი exit. ის გამოიყენება შეცდომების შემოწმებისას, მაგალითად, როცა შეკვეთა არ განხორციელდა , მაშინ შესაზლებელი იყო გამოტანილიყო შეტყობინება ”თქვენ არაფერი არ შრგიკვეთავს” და მერე exit ოპერატორით ფასრულებულიყო სცენარის დარჩენილი კოდის შესრულება, მაფალითა ეს ჩაიწერებოდა შემდეგნაირად:

ლისტინგი 10

if ($totalqty==0) {

echo '<font color="red"><h3 align="center">თქვენ არაფერი არ შეგიკვეთიათ წინა გვერდზე!</h3></font><br>';

exit;

}

მმართველი სტრუქტურების ალტერნატიული სინტაქსისი

ყველა ზემოთ განხილული მმართველი სტრუქტურებისთვის, ერთის გარდა არსებობს მათი ალტერნატიული სინტაქსის გამოყენება. კერძოდ, { - გამღები ფიგურული ფრჩხილი შეიძლება შეიცვალოს ორწერტილით (:); ხოლო დამხურავი ფიგურული ფრჩხილი - }, ახალი სამოსამსახურე სიტყვით, რომელიც შეიძლება იყოს ან endif, ან endswitch, ან endwhile,ან endfor ან endforeach - იმისდა მიხედვით, თუ რომელი მმართველი სტრუქტურა გამოიყენება.

მაგალითად ლისტინგი 10 შეძლება ჩაიწეროს შემდეგი სინტაქსისით:

ლისტინგი 11 ალტერნატიული სინტაქსისი

if ($totalqty==0) :

echo '<font color="red"><h3 align="center">თქვენ არაფერი არ შეგიკვეთიათ წინა გვერდზე!</h3></font><br>';

exit;

endif ;