

فصل ۳

کنترل روند برنامه

هدف‌های رفتاری:

پس از آموزش این فصل هنرجو می‌تواند:

- ۱- ساختار دستور `if...else` را بشناسد و در مواقع لزوم آن را در برنامه به کار ببرد.
- ۲- با دستور `switch` کار کند.
- ۳- با انواع حلقه‌های تکرار برنامه نویسی کند.

معمولاً تمام زبان‌های برنامه‌نویسی، دستورات مختلفی برای کنترل روند برنامه دارند. زبان PHP تقریباً تمام دستوراتی که در زبان‌های دیگر، برای انجام عملیات کنترلی برنامه در نظر گرفته شده است را پشتیبانی می‌کند.

دستورات کنترلی در زبان‌های برنامه‌نویسی به دو گروه اصلی تقسیم می‌شوند :

– دستورات کنترلی شرطی

– حلقه‌های تکرار

دستورات شرطی، روی جریان اجرای بخشی از برنامه تأثیر می‌گذارند. اما حلقه‌های تکرار تکه‌ای از برنامه را به تعداد دفعات مشخص، یا رسیدن به شرط خاصی اجرا می‌کنند. انواع دستورات شرطی و انواع حلقه‌های تکرار، برای اجرا به متغیرها نیاز دارند و برای اجرای آنها باید متغیرها شرایط خاصی داشته باشند. لذا به علت وابستگی اجرای دستورات به مقدار متغیرها، لازم است در طول برنامه مقادیر متغیرها ارزیابی شوند.

۱-۳- دستور if

یکی از کاربردی‌ترین دستورات زبان‌های برنامه‌نویسی، دستور شرطی if است. این دستور مفهومی شبیه به همان کلمه "if" در زبان انگلیسی و کلمه "اگر" در زبان فارسی دارد. عملکردی کاملاً مشابه جملات شرطی محاوره‌ای دارد. ساختار کلی دستور if به صورت زیر است :

if (عبارت شرطی)

{

دستورات بدنه در صورت برقراری عبارت شرطی

}

else

{

دستورات بدنه در صورت عدم برقراری عبارت شرطی

}

کد زیر یک برنامه ساده است که با دستور شرطی if نوشته شده است :

```
if ( $a == 0 ) {
```

```
echo " found a 0 ! " ;  
}
```

شرط فوق مقدار a را با صفر مقایسه می‌کند، اگر a برابر با صفر بود پیغامی به صورت "found a 0" را در مرورگر نمایش می‌دهد.

نکته: اگر دستورات بدنه که باید در صورت برقراری شرط اجرا شوند، از یک خط بیشتر نباشد می‌توان از نوشتن { } صرف‌نظر کرد.

۱-۱-۳- دستور else

یکی از قابلیت‌های دستور شرطی if این است که برنامه‌نویس می‌تواند معین کند که اگر شرط برقرار بود، یک عمل خاص و در غیر این صورت عمل دیگری را انجام دهد.

else به معنای "در غیر این صورت" در زبان انگلیسی با همان معنا در برنامه‌نویسی نیز به کار برده می‌شود.

شرط if مثال اول را طوری دوباره‌نویسی می‌کنیم که اگر مقدار a برابر با صفر نبود پیغام دیگری را نمایش دهد:

```
if ( $a == 0 )  
{  
echo " found a = 0 ! " ;  
}  
else  
{  
echo " not found ! " ;  
}
```

۲-۱-۳- دستور elseif

اگر در دستور شرطی if بخواهید یک مجموعه از شرط‌ها را مدیریت کنید، باید از دستور elseif استفاده کنید.

این دستور، مانند چند دستور شرطی مستقل، عمل می‌کند و در صورت برقراری هر کدام از شرط‌ها، کدهای مشخصی را برای اجرا انتخاب می‌کند.

برنامه‌زیر برای عددی که در متغیر \$num قرار دارد ضمن انجام عمل مقایسه توسط دستور if، مثبت، منفی و یا مساوی صفر بودن را تعیین می‌کند:

```

$num=0;
if ($num < 0)
{
echo "$num is negative";
}
elseif ($num == 0)
{
echo "$num is zero";
}
elseif ($num > 0)
{
echo "$num is positive";
}

```

مطالعه آزاد

همان‌طور که می‌دانید کدهای PHP را می‌توان با HTML ترکیب کرد، اگر بخواهید کد مثال فوق را به صورت ترکیب کدهای PHP و HTML بازنویسی کنید، می‌توانید از دستور echo صرف نظر کرده، تنها از پرچسب‌های HTML برای نمایش متن استفاده نمایید. در این صورت می‌توان کد مثال قبل را به صورت زیر بازنویسی کرد:

```

<?php
$num=0;
if ($num < 0): ?>
<h1><?php echo "$num";?> is negative</h1>
<?php elseif($num == 0): ?>
<h1><?php echo "$num";?> is zero</h1>
<?php elseif ($num > 0): ?>
<h1><?php echo "$num";?> is positive</h1>
<?php endif; ?>

```

در اینجا هنگام استفاده از این ساختار، به دلیل جدا شدن کدهای PHP از یکدیگر، می‌توان به جای { هنگام شروع بلوک کد، از علامت " : " استفاده نمود و در این صورت نیازی به تعیین مکان } برای خاتمه بلوک کد نیست و در انتها باید با استفاده از دستور endif تمام دستور شرطی معین شود.

۳-۱-۳ عملگر شرطی؟

یکی از قابلیت‌های جالب زبان PHP، وجود عملگر شرطی؛ است که از آن برای پیاده‌سازی دستورات شرطی می‌توان استفاده نمود. شکل کلی این دستور به صورت زیر است:

عملیات ۲: عملیات ۱؟ (عبارت شرطی)

عملیات ۲ در صورت غلط بودن عبارت شرطی و عملیات ۱، در صورت صحیح بودن عبارت شرطی انجام می‌شود.

مثال: در مثال زیر اگر مقدار نمره (\$num)، مساوی یا بیشتر از ۱۲ باشد، در خروجی رشته Pass (قبول) چاپ می‌شود و اگر کمتر از ۱۲ باشد، در خروجی رشته fail (به معنای مردود) چاپ می‌گردد.

```
<?php
$num =13;
$result=($num>=12)? "pass" : "fail";
print "$result";
?>
```

خروجی این مثال عبارت است از:

pass

۳-۲-۳ دستور switch

مثال زیر را با دستور if... else... در نظر بگیرید در این مثال به ازای مقادیر مختلف یک متغیر، دستورات متفاوتی اجرا می‌شود. یعنی در واقع چند شرط را برای یک متغیر بررسی می‌کنیم.

مثال:

```
$a=3;
if ( $a == 0 )
    echo " found a 0 ! " ;
elseif ( $a == 1 )
    echo " found a 1 ! " ;
elseif ( $a == 2 )
    echo " found a 2 ! " ;
elseif ( $a == 3 )
```

```
echo " found a 3 ! " ;
```

خروجی : found a 3!

اگر چه ساختار مثال فوق ساده است، اما در صورتی که تعداد شرط‌ها و تعداد دستوراتی که باید در صورت درستی هر شرط اجرا شوند زیاد باشد، ممکن است باعث پیچیدگی برنامه‌ها شود. برای چنین ساختارهایی در زبان‌های برنامه‌نویسی، از جمله PHP، دستور switch در نظر گرفته شده است. دستور if elseif مثال قبل با دستور switch به این صورت نوشته می‌شود.

مثال :

```
switch ( $a )
{
    case "0" :
echo " found a 0 ! " ;
    break;
case "1" :
    echo " found a 1 ! " ;
    break;
case "2" :
    echo " found a 2 ! " ;
    break;
case "3" :
    echo " found a 3 ! " ;
    break;
default:
    echo " not found ! " ;
}
```

اجزای دستور switch عبارتند از :

- **کلمه switch** : مقابل کلمه switch متغیر یا عبارتی که باید مقادیر مختلف آن بررسی شود قرار می‌گیرد.
- **کلمه case** : مقادیر مختلف و مرتبط با متغیر مقابل عبارت switch تعیین می‌شود.
- **دستور default** : برای تعیین دستورات پیش فرض ساختار switch استفاده می‌شود. در

واقع توسط این دستور می‌توان تعیین کرد که در صورت عدم برقراری تمام مقادیر موجود در case ها، چه عملی انجام شود.

نکته: در هر case از برنامه، برای جلوگیری از اجرای شدن case بعدی، باید از دستور break برای خروج از آن قسمت از برنامه استفاده کرد. اگر درون یک دستور case از break استفاده نشود، دستور مربوط به case بعدی اجرا می‌شود.

مثال: برنامه زیر محدوده نمره دانش‌آموز را بررسی می‌کند چنانچه بین صفر و کمتر از ۱۰ باشد رتبه F و اگر بین ۱۰ تا ۱۲ باشد رتبه E و اگر بیشتر از ۱۲ تا ۱۴ باشد رتبه D و ... را برای دانش‌آموز چاپ می‌کند.

```
$num = 15;
switch ($num)
{
case (($num<10)&& ($num>=0)):
    echo "your grade is F";
    break;
case (($num>=10)&& ($num<=12)):
    echo "your grade is E";
    break;
case (($num>12)&& ($num<=14)):
    echo "your grade is D";
    break;
case (($num>14)&& ($num<=16)):
    echo "your grade is C";
    break;
case (($num>16)&& ($num<=18)):
    echo "your grade is B";
    break;
case (($num>18)&& ($num<=20)):
    echo "your grade is A";
    break;
default:
    echo "It is a wrong number.";
}
```

خروجی :

your grade is C

۳-۳- حلقه‌های تکرار

فرض کنید می‌خواهیم یک عبارت را سه بار در خروجی نمایش دهیم، این برنامه را به صورت زیر خواهیم نوشت :

```
echo " hello " ;
```

```
echo " hello " ;
```

```
echo " hello " ;
```

حال فرض کنید همان عبارت را می‌خواهیم ۱۰۰ بار در خروجی نمایش دهیم، احتمالاً خواهید گفت صد بار یک خط از برنامه فوق را می‌نویسیم. اما در زبان‌های برنامه‌نویسی امکاناتی وجود دارد که به برنامه‌نویس کمک می‌کند بدون نیاز به کار تکراری، یک مجموعه عملیات را به تعداد دفعات مورد نیاز انجام دهد. به این ساختارها "حلقه" گفته می‌شود.

مفهوم حلقه : حلقه‌ها برای اجرای یک یا چند دستورالعمل به تعداد دفعات مشخص یا تا زمان رسیدن به شرط خاصی به کار برده می‌شوند. حلقه‌های زبان PHP دارای دو نوع کلی `for` و `while` هستند.

۱-۳-۳- حلقه `for`

این حلقه زمانی کاربرد دارد که می‌خواهیم مجموعه‌ای از عملیات، به تعداد دفعات معینی انجام شوند. ساختار حلقه `for` به صورت زیر است :

(میزان افزایش یا کاهش متغیر شمارنده حلقه؛ عبارت کنترلی؛ مقدار اولیه متغیر) `for`

```
{  
دستورات بدنه حلقه  
}
```

- **مقدار اولیه متغیر :** مقدار اولیه برای شروع اجرای حلقه را مشخص می‌کند.
- **عبارت کنترلی :** در این قسمت یک عبارت شرطی درج می‌شود که در هر بار اجرای حلقه مقدار متغیر شمارنده حلقه توسط این عبارت کنترلی که در واقع شرط انجام حلقه است، چک می‌شود، اگر جواب شرط `True` بود، اجرای حلقه ادامه می‌یابد و در غیر این صورت اجرای حلقه پایان می‌پذیرد.
- **میزان افزایش یا کاهش :** مقدار متغیر شمارنده حلقه را تعیین می‌کند که در هر بار اجرای دستورات مقدار این متغیر افزایش یا کاهش یابد.

نکته : از میان سه پارامتر اصلی ساختار حلقه `for`، نوشتن مقدار اولیه و میزان افزایش یا کاهش اختیاری است اما نوشتن عبارت کنترلی در ساختار `for` اجباری است.

در صورتی که میزان افزایش یا کاهش حلقه مقابل for نوشته نشود، می‌توان آن را در بدنه حلقه تعیین کرد و افزایش یا کاهش داد. مقدار اولیه را نیز می‌توان قبل از شروع حلقه for مقداردهی کرد.

```
<?php
for ( $n = 2 ; $n <= 5 ; $n++ )
{
    echo "number: " . $n . "<br/>";
}
?>
```

خروجی :

```
number: 2
number: 3
number : 4
number: 5
```

در مثال فوق مقدار اولیه متغیر \$n برابر با ۲ تعریف شده است و شرط خاتمه حلقه توسط عبارت کنترلی $n \leq 5$ تعیین شده است، مقدار \$n نیز در هر مرحله، ۱ واحد افزوده می‌شود.

مثال:

کد زیر اعداد ۱ تا ۳ را در خروجی چاپ می‌کند.

```
<?php
for ( $n = 1 ; $n <= 5 ; $n++ )
{
    echo("number: " . $n . "<br/>");
    if ( $n == 3 ) break ;
}
?>
```

خروجی :

```
number: 1
number: 2
```

number: 3

نکته: ممکن است گاهی اوقات هنگام اجرای حلقه، تحت شرط یا شرایط خاصی بخواهید از آن خارج شده و ادامه اجرا را متوقف نمایید. در این شرایط می‌توانید از دستور break استفاده کنید. مانند آنچه در مثال فوق انجام شد.

مثال:

کد زیر مجذور اعداد کوچک‌تر از ۱۰ را نمایش می‌دهد.

```
<?php
for ($i = 0; $i < 10; $i++) {
print "The square of $i is " . $i*$i . "<br/>";
}
?>
```

خروجی:

The square of 0 is 0

The square of 1 is 1

The square of 2 is 4

The square of 3 is 9

The square of 4 is 16

The square of 5 is 25

The square of 6 is 36

The square of 7 is 49

The square of 8 is 64

The square of 9 is 81

نکته: علاوه بر دستور break، دستور continue نیز در حلقه‌ها کاربرد دارد. دستور continue ادامه اجرای حلقه با مقدار شمارنده فعلی را متوقف می‌کند و اجرای حلقه را با مقدار شمارنده بعدی از سر می‌گیرد. برای فهم بهتر به مثال بعدی توجه کنید.

نکته: تأثیر break و continue بر دستورات بعد از خودشان است و دستورات قبل از آنها به طور کامل اجرا می‌شوند.

مثال:

```
<?php
```

```

for ( $n = 1 ; $n <= 5 ; $n++ )
{
    if ( $n == 2 ) continue ;
    echo("number: ".$n."<br />");
}
?>

```

خروجی :

```

number: 1
number: 3
number: 4
number: 5

```

نکته : برای چاپ چند عبارت یا متغیر توسط دستور echo باید عبارات یا متغیرهای مورد نظر توسط علامت . به یکدیگر چسبانده شوند . به عنوان مثال ;echo("number: ".\$n); عبارت number: و به دنبال آن مقدار متغیر \$n را چاپ می کند .

نکته : در دستور echo درج عبارت "
" باعث می شود که مکان نما پس از چاپ عبارت موجود، به ابتدای سطر بعد منتقل شود .

مثال :

چاپ اعداد ۵ تا ۲ با مقداردهی اولیه خارج از حلقه به صورت زیر است :

```

<?php
$n=5 ;
for ( ; $n >= 2 ; $n -- )
{
    echo("number: " . $n . "<br/>");
}
?>

```

خروجی :

```

number: 5
number: 4
number: 3
number: 2

```

مثال: مثال فوق با انتقال عمل کاهش متغیر شمارنده به داخل بدنه حلقه به صورت زیر است.

```
<?php
$n=5 ;
for ( ; $n >= 2 ; )
{
    echo("number: " . $n . "<br/>");
    $n-- ;
}
?>
```

خروجی :

```
number: 5
number : 4
number: 3
number: 2
```

۲-۳-۳- حلقه foreach

این نوع از حلقه برای کار با عناصر آرایه در نظر گرفته شده است. ساختار کلی حلقه foreach به صورت زیر است :

```
foreach ($array as نام متغیر)
{
    بدنه حلقه و عباراتی که باید اجرا شوند
}
```

هنگام اجرای این حلقه، عناصر آرایه از ابتدا تا انتها یکی یکی در متغیری که به عنوان ورودی Foreach ذکر شده، قرار می گیرند در مثال زیر \$value است. نام متغیر \$value می تواند هر نام دلخواه برنامه نویس باشد. برای آشنایی با عملکرد این حلقه به مثال زیر توجه کنید :

مثال:

```
<?php
$x=array("one","two","three");
```

```
foreach ($x as $value)
```

```
{  
echo $value . "<br />";
```

```
    {  
?>
```

در اولین اجرای حلقه، اولین مقدار آرایه در متغیر \$value قرار می‌گیرد و به دلیل اینکه دستور بدنه foreach، چاپ این متغیر و سپس رفتن به ابتدای سطر بعد است، خروجی به صورت زیر خواهد بود:

```
one  
two  
three
```

۳-۳-۳- حلقه while

حلقه while نوعی حلقه تکرار است و زمانی از آن استفاده می‌شود که بخواهیم عملیات مشخصی، تا زمانی که شرط تعیین شده‌ای برقرار باشد، اجرا شود. ساختار حلقه while به این صورت است:

```
while (شرط حلقه)  
{  
    دستورات بدنه حلقه  
    به روز رسانی متغیر  
}
```

مثال:

برنامه زیر با استفاده از حلقه while اعداد ۲ تا ۵ را نمایش می‌دهد:

```
<?php  
$n=2 ;  
while ( $n <= 5 )  
{  
echo("number: " . $n. "<br/>");  
$n++;  
}  
?>
```

خروجی :

number: 2

number: 3

number: 4

number: 5

تمرین: با استفاده از حلقه تکرار while، اعداد زوج کوچک‌تر از ۱۰ را نمایش دهید.

۴-۳-۳- حلقه do ... while

ساختار حلقه do ... while مانند حلقه while است، تنها تفاوت این دو نوع حلقه در این است که حلقه do ... while شرط را در انتها کنترل می‌کند، یعنی، پس از هر بار اجرای دستورات حلقه شرط در انتها بررسی می‌شود و اگر شرط برقرار باشد دفعه بعد نیز اجرای حلقه از سر گرفته می‌شود، اما اگر شرط برقرار نباشد، بدون بازگشت به ابتدای حلقه، از حلقه while خارج می‌شوند.

نکته: در حلقه while در صورت عدم برقراری شرط، دستورات بدنه حلقه هرگز اجرا نمی‌شود اما در do ... while حتی در صورت برقرار نبودن شرط دستورات بدنه، حلقه حداقل یک بار اجرا می‌شود.

ساختار حلقه do ... while به صورت زیر است :

```
do
{
دستورات بدنه حلقه
}
while (شرط حلقه)
```

مثال:

برنامه زیر با استفاده از حلقه do اعداد ۲ تا ۵ را نمایش می‌دهد :

```
<?php
$n=2 ;
do
{
    echo("number: " . $n. "<br/>");
    $n++;
}
while( $n<=5)
?>
```

خروجی :

number: 2

number: 3

number: 4

number: 5

مثال: در کد زیر از یک حلقه do...while استفاده شده است که شرط خاتمه آن این است

که متغیر \$n کوچک تر یا مساوی ۳ باشد، اما همان طور که در کد ملاحظه می کنید، این متغیر قبل از شروع حلقه با عدد ۵ مقداردهی شده است. یعنی با وجود برقرار نبودن شرط اجرای حلقه، عملیات بدنه while یک بار اجرا می شود.

```
<?php
$n = 5 ;
do
{
    echo(" number : " . $n . "<br />");
    $n++;
}
while ( $n <= 3 )
?>
```

number : 5

خروجی :

تمرین: با استفاده از حلقه تکرار do ... while اعداد زوج کوچک تر از ۱۰۰ را نمایش دهید.

دستورات کنترلی در زبان‌های برنامه نویسی به دو گروه اصلی تقسیم می‌شوند:

– دستورات کنترلی شرطی

– حلقه‌های تکرار

دستورات شرطی روی جریان اجرای بخشی از برنامه تأثیر می‌گذارند اما حلقه‌های تکرار تکه‌ای

از برنامه را به تعداد دفعات مشخص یا تا زمان رسیدن به شرطی خاص، اجرا می‌کنند.

یکی از کاربردی‌ترین دستورات زبان‌های برنامه نویسی، دستور شرطی `if` است که عملکردی

کاملاً مشابه جملات شرطی محاوره‌ای دارند.

در دستور `switch` از `default` برای تعیین دستورات پیش‌فرض ساختار `switch` استفاده

می‌شود.

حلقه‌های زبان PHP دارای دو نوع کلی `for` و `while` هستند.

از حلقه `while` زمانی استفاده می‌شود که بخواهیم عملیات مشخصی، تا زمانی که شرط تعیین

شده‌ای برقرار باشد، اجرا شود.

حلقه `for` زمانی کاربرد دارد که می‌خواهیم مجموعه‌ای از عملیات به تعداد دفعات معینی

انجام شوند.

- ۱- برنامه‌ای بنویسید که اعداد کوچک‌تر از 100 و بزرگ‌تر از 20 را به همراه زوج یا فرد بودن آن با استفاده از دستورات `for` و `if`، چاپ کند.
 - ۲- زوج یا فرد بودن اعداد در سؤال اول را با دستور `switch` به جای `if` بررسی کنید.
 - ۳- برنامه سؤال اول را با استفاده از دستورات `do ... while` و `if` بازنویسی کنید.
 - ۴- یک آرایه 5 عنصری برای معرفی رنگ‌های مورد علاقه خود تعریف کنید، سپس با استفاده از دستور `foreach` آنها را در مرورگر نمایش دهید.
 - ۵- برنامه‌ای بنویسید که مجموع ارقام یک عدد مفروض را چاپ نماید.
 - ۶- برنامه‌ای بنویسید که براساس شماره‌های 1 تا 4 به ترتیب چهار عمل اصلی `+`، `-`، `÷` و `×` را روی دو عدد مفروض انجام دهد.
- یعنی اگر متغیر مورد نظر مقدار 1 داشت عمل جمع و اگر مقدارش 2 بود عمل تفریق و ... بر روی دو عدد دلخواه انجام گیرد.