

Web Programming

Chapter 3.1. Conditional Statements

Objectives

- ◆ To learn to use conditional test statements to compare numerical and string data values
- ◆ To learn to use looping statements to repeat statements
- ◆ To learn to use logical test operators to create compound conditional test statements

Content

1. Using Conditional Test Statements
2. Using Loops to Repeat Statements

Content

- 1. Using Conditional Test Statements
- 2. Using Loops to Repeat Statements

1. Conditional Test Statements

- ◆ Conditional statements provide a way for scripts to test for certain data values and then to react differently depending on the value found.
- ◆ Will examine
 - the if statement,
 - the elseif clause,
 - the else clause,
 - and the switch statement.

1.1. Using the if Statement

- ◆ Use an if statement to specify a test condition and a set of statements to run when a test condition is *true*.

```
if ($average > 69) {  
    $Grade="Pass";  
    print "Grade=$Grade ";  
}  
print "Your average was $average";
```

When \$average is greater than
69 execute these statements.

- ◆ if \$average was equal to 70 then the above would output:

Your average was 70

a. Test Expressions

- ◆ Test expressions use test operators within their expressions.
 - Test operators work much like the expression operators.
 - The if statement above uses the greater than (>) operator to test whether \$average is greater than 69.
 - Test operators evaluate to *true* or *false*

PHP Test Operators

Operator Test	Effect	Example	Result
<code>==</code>	Equal to	<code>if (\$x == 6){ \$x = \$y + 1; \$y = \$x + 1; }</code>	Run the second and third statements if the value of <code>\$x</code> <i>is equal to</i> 6.
<code>!=</code>	Not equal to	<code>if (\$x != \$y) { \$x = 5 + 1; }</code>	Run the second statement if the value of <code>\$x</code> <i>is not equal to</i> the value of <code>\$y</code> .
<code><</code>	Less than	<code>if (\$x < 100) { \$y = 5; }</code>	Run the second statement if the value of <code>\$x</code> <i>is less than</i> 100.
<code>></code>	Greater than	<code>if (\$x > 51) { print "OK"; }</code>	Run the second statement if the value of <code>\$x</code> <i>is greater than</i> 51.
<code>>=</code>	Greater than or equal to	<code>if (16 >= \$x) { print "x=\$x"; }</code>	Run the second statement if 16 <i>is greater than or equal to</i> the value of <code>\$x</code> .
<code><=</code>	Less than or equal to	<code>if (16 >= \$x) { print "x=\$x"; }</code>	Run the second and third statements if the value of <code>\$x</code> <i>is less than or equal to</i> the value of <code>\$y</code> .

A Full Example ...

- ◆ Consider the following application:
 - Receives two grades as input and determines whether their average is above 89.
 - It uses an HTML form for input grades:

```
Enter First Score <input type="text" size="4"  
maxlength="7" name="grade1">
```

```
Enter Second Score <input type="text" size="4"  
maxlength="7" name="grade2">
```

Sets
\$grade1

Sets
\$grade2

Receiving Code

```
1. <html>
2. <head><title>Decisions</title></head>
3. <body>
4. <?php
5.     $average = ($grade1 + $grade2) / 2;           Calculate average
6.     if ( $average > 89 ) {
7.         print "Average score: $average You got an A! <br>";
8.     }
9.     $max=$grade1;
10.    if ($grade1 < $grade2) {
11.        $max = $grade2;                            Output if $average
12.    }                                              is more than 89
13.    print ("Your max score was $max");
14. ?>
15. </body></html>
```

Receiving Code With REGISTER_GLOBALS off

```
1. <html>
2. <head><title>Decisions</title></head>
3. <body>
4. <?php
5.     $grade1= $_POST["grade1"];
6.     $grade2= $_POST["grade2"];
5.     $average = ($grade1 + $grade2) / 2;
6.     if ( $average > 89 ) {
7.         print "Average score: $average You got an A! <br>";
8.     }
9.     $max=$grade1;
10.    if ($grade1 < $grade2) {
11.        $max = $grade2;
12.    }
13.    print ("Your max score was $max");
14. ?>
15. </body></html>
```

Get grade1 and grade2 from HTML form.

Calculate average

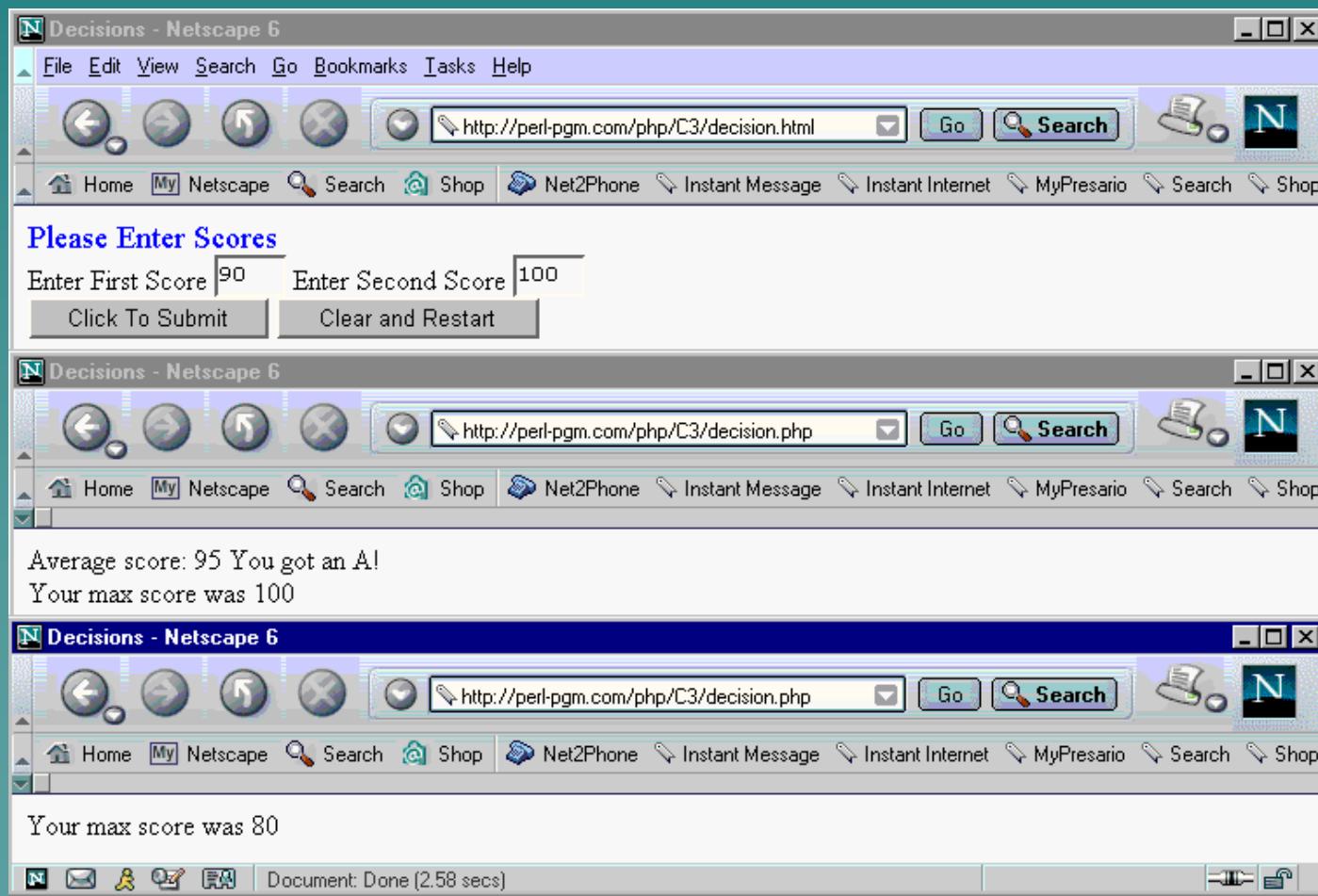
Output if \$average is more than 89

Set when \$grade2 is more than \$grade1

A Full Example ...

The previous code can be executed at

<http://webwizard.aw.com/~phppgm/C3/decision.html>



b. Comparing Strings

- ◆ PHP represents strings using the ASCII code values (American Standard Code for Information Interchange).
 - ASCII provides a standard, numerical way to represent characters on a computer.
 - Every letter, number, and symbol is translated into a code number.
 - ◆ "A" is ASCII code 65, "B" is 66, "C" is 67, and so on.
 - ◆ Lowercase "a" is ASCII code 97, "b" is 98, "c" is 99, and so on.
 - ◆ ASCII "A" is less than ASCII "a," "B" is less than "b," and "c" is less than "d".
 - ◆ ASCII characters have ASCII code values lower than letters. So ASCII character "1" is less than "a" or "A"

b. Comparing Strings (2)

- ◆ You can use == operator to check if one string is equal to another. For example,

```
$name1 = "George"; $name2 = "Martha";
if ($name1 == $name2) {
    print ("$name1 is equal to $name2" );
} else {
    print ("$name1 is not equal to $name2");
}
```

- ◆ Would output: “George is not equal to Martha”.

b. Comparing Strings (3)

- ◆ Also can use <, >, <=, and >= operators to compare string values using ASCII code values.
- ◆ For Example

```
$name1 = "George"; $name2 = "Martha";  
if ($name1 < $name2) {  
    print ("$name1 is less than $name2");  
} else {  
    print ("$name1 is not less than $name2");  
}
```

- ◆ It would output “George is less than Martha”.

A Full Example ...

- ◆ Consider the following application:
 - Compares two input strings.
 - It uses the HTML form element that sets the variables \$first and \$second.

First Name: <input type="text" size="10"
 maxlength="15" name="first">

Sets
\$first

Second Name: <input type="text" size="10"
 maxlength="15" name="second">

Sets
\$second

Receiving Code

```
1. <html>
2. <head><title>String Comparison Results</title></head>
3. <body>
4. <?php
5. print ("First=$first Second=$second<br>");
6. if ($first == $second) {
7.   print ("$first and $second are equal");
8. }
9. if ($first < $second) {
10.   print ("$first is less than $second");
11. }
12. if ($first > $second) {
13.   print ("$first is greater than $second");
14. }
15. ?></body></html>
```

The diagram illustrates the logic of the PHP script. Three green callout boxes with white text provide explanations for different branches of the if-else-if-else structure:

- A box labeled "Output if \$first is equal to \$second" points to line 7, which prints the strings if the two variables are equal.
- A box labeled "Set when \$second is less than \$first" points to line 10, which prints the strings if \$first is less than \$second.
- A box labeled "Set when \$first is more than \$second" points to line 13, which prints the strings if \$first is greater than \$second.

Receiving Code With REGISTER_GLOBALS OFF

```
1. <html>
2. <head><title>String Comparison Results</title></head>
3. <body>
4. <?php
5. $first = $_POST["first"];
6. $second = $_POST["second"];
7. print ("First=$first Second=$second<br>");
8. if ($first == $second) {
9.   print ("$first and $second are equal");
10. }
11. if ($first < $second) {
12.   print ("$first is less than $second");
13. }
14. if ($first > $second) {
15.   print ("$first is greater than $second");
16. }
17. ?></body></html>
```

Get the values of \$first and \$second

Output if \$first is equal to \$second

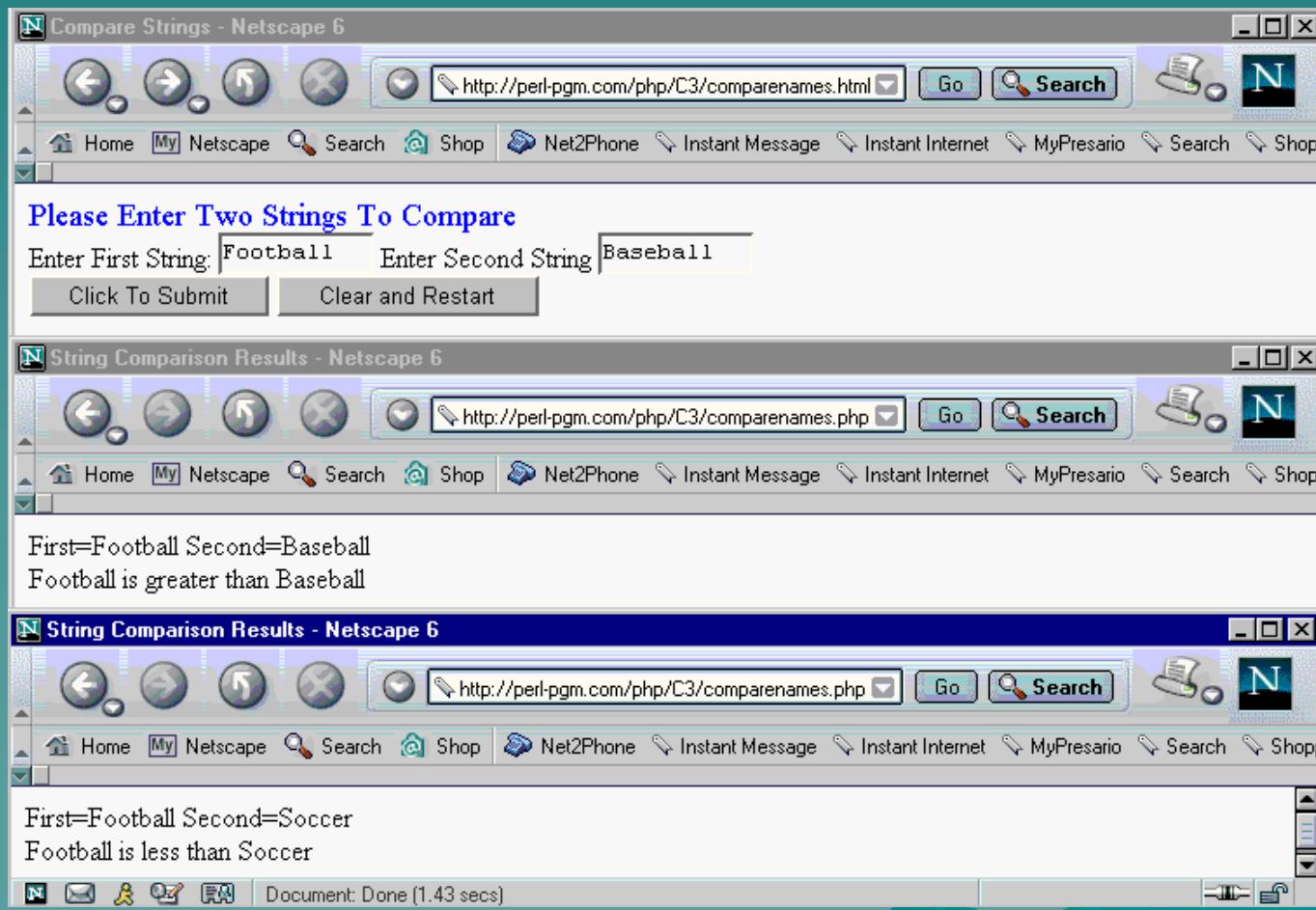
Set when \$second is less than \$first

Set when \$first is more than \$second

The Output ...

The previous code can be executed at

<http://webwizard.aw.com/~phppgm/C3/comparenames.html>



c. Using the elseif Clause

- ◆ Use an elseif clause with an if statement to specify an additional test condition

```
if (test expression) {  
    one or more PHP statements  
} elseif (test expression)  
    one or more PHP statements  
}
```

- ◆ The above script checks the elseif test expression when the test condition for the if statement is *false*.

c. Using the elseif Clause (2)

- ◆ One or more elseif clauses can be used with an if statement.

```
if ($hour < 9) {  
    print "Sorry, it is too early.";  
} elseif ($hour < 12) {  
    print "Good morning. The hour is $hour. ";  
    print "How can we help you?";  
} elseif ($hour < 13) {  
    print "Sorry, we are out to lunch. ";  
} elseif ($hour < 17) {  
    print "Good afternoon. The hour is $hour. ";  
    print "How can we help you?";  
} elseif ($hour <= 23) {  
    print "Sorry, we have gone home already.";
```

Check this test expression when the first condition is *false*.

Check this test expression when the first two conditions are all *false*.

Check this test expression when the first three conditions are all *false*.

if \$hour == 15, output “Good afternoon. The hour is 15. How can we help you?” if \$hour == 24, then this code outputs nothing.

d. Using the else Clause

- ◆ Use an else clause with if and possibly one or more elseif clauses

- Specify set of statements to run when all the previous test conditions are *false*.
 - Has the following general format shown in the

```
if (test expression) {  
    one or more PHP statements  
} else {  
    one or more PHP statements  
}
```

d. Using the else Clause (2)

- ◆ For example, if \$count had a value of -75, then this code would output “Illegal value for count = -75”

```
if ( $count == 0 ) {  
    print ("Time to reorder.");  
    $reorder=1;  
} elseif ( $count == 1 ) {  
    $reorder=1;  
    print ("Warning: we need to start reordering.");  
} elseif ( $count > 1 ) {  
    $reorder = 0;  
    print ("We are OK for now.");  
} else {  
    print ("Illegal value for count = $count");  
}
```

A Full Example ...

- ◆ Full example that extends the grade-averaging to determine a letter grade (A, B, C, D, or F) and to catch illegal input.
- ◆ Use the following HTML form for input

```
Enter First Score <input type="text" size="4"  
maxlength="7" name="grade1">
```

Sets
\$grade1

```
Enter Second Score <input type="text" size="4"  
maxlength="7" name="grade2">
```

Sets
\$grade2

Receiving Code

```
1. <html>
2. <head><title>Grade Calculation</title></head>
3. <body>
4. <?php
5. $average = ($grade1 + $grade2) / 2; ← Compute average of
6. if ($average > 89) { $grade1 and $grade2
7.     print ("Average=$average You got an A");
8. } elseif ($average > 79) { ← Check if $average
9.     print ("Average=$average You got a B");
10. } elseif ($average > 69) { ← is an "A", "B", "C",
11.     print ("Average=$average You got a C");
12. } elseif ($average > 59) { ← "D" or "F"
13.     print ("Average=$average You got a D");
14. } elseif ($average >= 0) { ←
15.     print ("Grade=$grade You got an F");
16. } else {
17.     print ("Illegal average less than 0 average=$average");
18. }
19. $max=$grade1;
20. if ($grade1 < $grade2) {
21.     $max = $grade2;
22. }
23. print ("<br>Your max score was $max");
24. ?> </body></html>
```

Compute average of
\$grade1 and \$grade2

Check if \$average
is an "A", "B", "C",
"D" or "F"

Receiving Code With REGISTER_GLOBALS Off

```
1. <html> <head><title>Grade Calculation</title></head>
2. <body>
3. <?php
4. $grade1 = $_POST["grade1"]; $grade2 = $_POST["grade2"];
5. $average = ($grade1 + $grade2) / 2;
6. if ($average > 89) {
7.     print ("Average=$average You got an A");
8. } elseif ($average > 79) {
9.     print ("Average=$average You got a B");
10. } elseif ($average > 69) {
11.     print ("Average=$average You got a C");
12. } elseif ($average > 59) {
13.     print ("Average=$average You got a D");
14. } elseif ($average >= 0) {
15.     print ("Grade=$grade You got an F");
16. } else {
17.     print ("Illegal average less than 0 average=$average");
18. }
19. $max=$grade1;
20. if ($grade1 < $grade2) {
21.     $max = $grade2;
22. }
23. print ("<br>Your max score was $max");
24. ?> </body></html>
```

Get values of \$grade1 and \$grade2

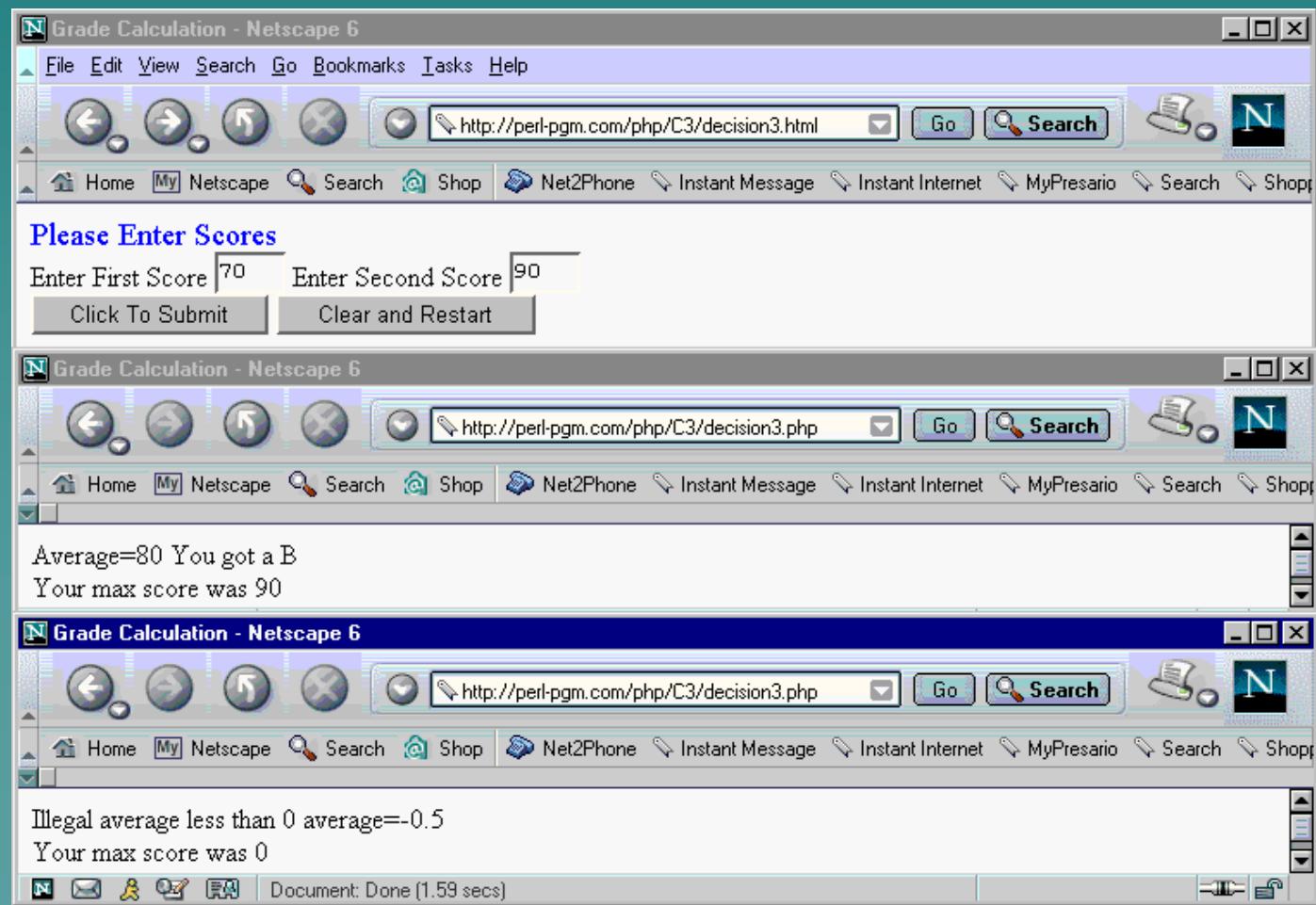
Compute average of \$grade1 and \$grade2

Check if \$average is an "A", "B", "C", "D" or "F"

Would output the following...

The previous code can be executed at

<http://webwizard.aw.com/~phppgm/C3/decision.php>



1.2. Using the switch Statement

◆ Use switch statement as another conditional test

```
1. switch ($rating) { ← Enclose in curly brackets  
2.   case 1:  
3.     $rated = "Poor"; ← Run these when $rating has  
4.     print "The rating was $rated";  
5.     break;  
6.   case 2:  
7.     $rated = "Fair"; ← Run these when $rating has  
8.     print "The rating was $rated";  
9.     break;  
10.  case 3:  
11.    $rated = "Good"; ← Run these when $rating has  
12.    print "The rating was $rated";  
13.    break;  
14.  default: ← When value not 1, 2, or 3.  
15.    print "Error: that rating does not exist";  
16. }
```

Content

1. Using Conditional Test Statements
- 2. Using Loops to Repeat Statements

2. Using Loops to Repeat Statements

- ◆ Scripts can use loop statements to repeat sections of code
- ◆ Advantages of loops include
 - Scripts can be more concise
 - Can write more flexible scripts
- ◆ Will discuss while loops and for loops now
 - Will review foreach loops later

2.1. Using a for loop

- ◆ Use a `for` loop to repeat of set of statements a specific number of times.

The *initialization expression* sets the initial value of \$i.
Enclose statements to repeat in curly brackets.

The *iteration expression* increments \$i at the end of each loop iteration.

The *loop-end condition* determines when the loop will end.

```
for ( $i = 0; $i < $max; $i++ ) {  
    Set of statements to repeat  
}
```

Note the use of ; after first 2 but not 3rd.

Full Script Example ...

```
1. <html><head><title>Loops</title></head>
2. <body><font size="5" color="blue">
3. Generate Square and Cube Values </font>
4. <br>
5. <form action="http://webwizard.aw.com/~phppgm/C3/whileloop.php" method="post">
6. <?php
7. print ("Select Start Number");
8. print ("<select name=\"start\">");
9. for ($i=0; $i<10; $i++) { ←
10.    print ("<option>$i</option>"); ←
11. }
12. print ("</select>"); ←
13. print ("<br>Select End Number");
14. print ("<select name=\"end\">"); ←
15. for ($i=10; $i<20; $i++) { ←
16.    print "<option>$i</option>"; ←
17. }
18. print ("</select>"); ←
19. ?>
20. <br><input type="submit" value="Submit">
21. <input type="reset" value="Clear and Restart"> </form></body></html>
```

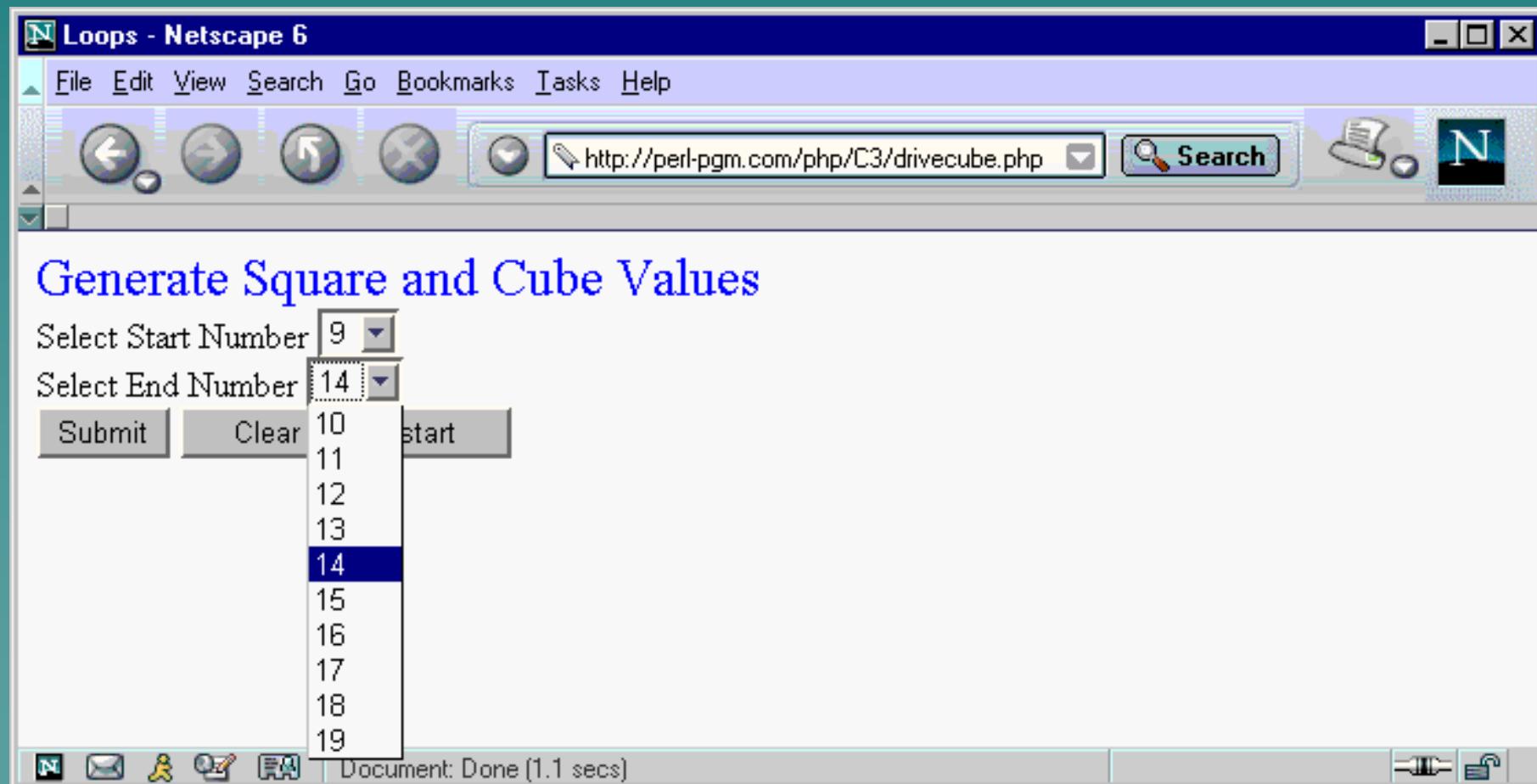
Repeat print statement
10 times with values 0,
1, 2, ... 9 for \$i.

Repeat print statement
10 times with values 10,
11, 12, ... 19 for \$i.

Would output the following...

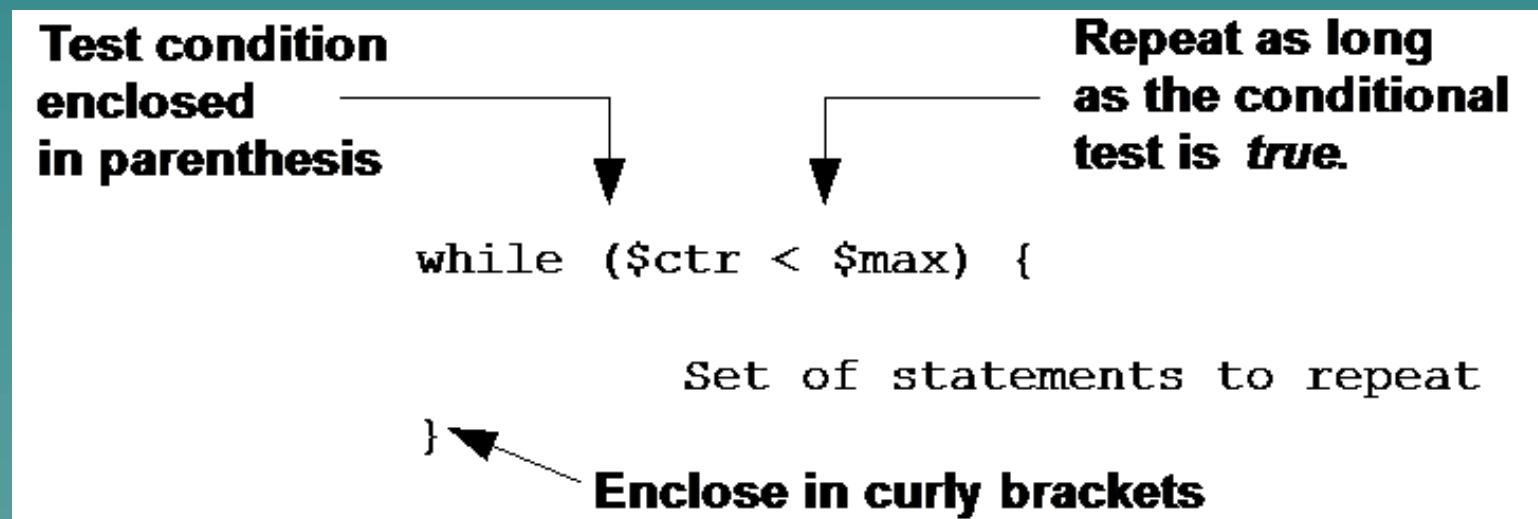
The previous code can be executed at

<http://webwizard.aw.com/~phppgm/C3/drivecube.php>



2.2. Using the while loop

- ◆ Use the while loop to repeat a set of statements as long as a conditional test is true.



2.2. Using the while loop (2)

- ◆ A while loop will repeat as long as the loop conditional test is *true*.
 - If initially *false*, then the statements within the loop body will never run.
- ◆ A bad idea to create an Infinite Loop
 - If the loop conditional test always *true*, then the loop will never end (infinite loop).
 - It will consume resources on the Web server and possibly slow down other server activity. (might have to exit the window that's running your script)

A Full Script Example ...

```
1. <html>
2. <head><title>While Loop</title></head>
3. <body>
4. <font size="4" color="blue"> Table of Square and Cube Values
</font>
5. <table border=1>
6. <th> Numb </th> <th> Sqr </th> <th> Cubed </th>
7. <?php
8.     $i = $start;
9.     while ($i <= $end) {
10.         $sqr=$i*$i;
11.         $cubed=$i*$i*$i;
12.         print ("<tr><td>$i</td><td>$sqr</td><td>$cubed</td></tr>");
13.         $i = $i + 1;
14.     }
15.?></table></body></html>
```

A Full Script Example (with REGISTER_GLOBALS off)

```
1. <html>
2. <head><title>While Loop</title></head>
3. <body>
4. <font size="4" color="blue"> Table of Square and Cube Values
</font>
5. <table border=1>
6. <th> Numb </th> <th> Sqr </th> <th> Cubed </th>
7. <?php
8.     $start = $_POST["start"];    $end = $_POST["end"];
9.     $i = $start;
10.    while ($i <= $end) {
11.        $sqr=$i*$i;
12.        $cubed=$i*$i*$i;
13.        print ("<tr><td>$i</td><td>$sqr</td><td>$cubed</td></tr>") ;
14.        $i = $i + 1;
15.    }
16.?></table></body></html>
```

The Output ...

The previous code can be executed at

<http://webwizard.aw.com/~phppgm/C3/whileloop.php>

A screenshot of the Netscape 6 browser window. The title bar reads "While Loop - Netscape 6". The address bar shows the URL "http://perl-pgm.com/php/C3/whileloop.php". The main content area displays a table titled "Table of Square and Cube Values". The table has three columns: "Numb", "Sqr", and "Cubed". The data rows are:

Numb	Sqr	Cubed
9	81	729
10	100	1000
11	121	1331
12	144	1728
13	169	2197
14	196	2744

At the bottom of the browser window, the status bar says "Document: Done (6.32 secs)".

TIP Using Either the while Loop or the for Loop for Some Problems

- ◆ For some loops you can use either the while loop or the for loop.

```
- for ( $i=0; $i<5; $i++ ) {  
    print "i=$i ";  
}  
- $i = 0;  
while ($i < 5) {  
    print "i=$i "; $i=$i + 1;  
}
```

The two above loops both output “i=0 i=1 i=2 i=3 i=4”.

2.3. Using Logical Test Operators

- ◆ PHP supports a set of logical test operators you can use to create compound test expressions
 - used within an if statement or a while statement to specify more than one test condition.
 - For example, consider the following line

```
while ($x > $max && $found != 1) {  
    . . .  
}
```

Logical Test Operators

- ◆ PHP supports three logical test operators.
 - `&&`: *the AND operator*
 - `||`: *the OR operator*
 - `!`: *the NOT operator*

And Operator

- ◆ Use in if statements and while loops.
- ◆ E.g.:

```
while ($ctr < $max && $flag == 0) {  
    ...  
}
```



Whenever either of
these expressions is *false*,
the loop will terminate.

Or operator

- ◆ Used much like the AND operator in if statements and while loops.
- ◆ E.g.

```
- if ($ctr != $max || $flag == 0) {
```

Carries out the statements within the if statement if either \$ctr is not equal to \$max or \$flag is equal to 0.

Not operator

- ◆ Used to test whether an expression is *false* (used in while loops and in if statements).
- ◆ E.g.

```
- if (!$flag == 0) {
```

This statement is *true* when
\$flag is anything except 0.

Example

- ◆ Asks the user to guess a “secret” two-digit combination, uses logical test operators.
- ◆ The Input HTML form uses the following to set pick1. A similar group sets a variable pick2.
 - ◆ Pick a number from 1 to 9

 - ◆ <input type="radio" name="pick1" value="1">1
 - ◆ <input type="radio" name="pick1" value="2">2
 - ◆ <input type="radio" name="pick1" value="3">3
 - ◆ <input type="radio" name="pick1" value="4">4
 - ◆ <input type="radio" name="pick1" value="5">5
 - ◆ <input type="radio" name="pick1" value="6">6
 - ◆ <input type="radio" name="pick1" value="7">7
 - ◆ <input type="radio" name="pick1" value="8">8
 - ◆ <input type="radio" name="pick1" value="9">9

A Full Script Example ...

```
1. <html><head><title>Number Guess Results </title></head>
2. <body>
3. <?php
4. $combo1=5;
5. $combo2=6;
6. if (($pick1 == $combo1) && ($pick2 == $combo2)) {
7.     print ("Congratulations you got both secret numbers
$combo1 $combo2!");
8. } elseif (($pick1 == $combo1) || ($pick2 == $combo2)) {
9.     print ("You got one number right.");
10. } else {
11.     print ("Sorry, you are totally wrong!");
12. }
13. print ("You guessed $pick1 and $pick2.");
14. ?></body></html>
```

A Full Script Example ... with REGISTER_GLOBALS off

```
1. <html><head><title>Number Guess Results </title></head>
2. <body>
3. <?php
4. $pick1 =$_POST["pick1"]; $pick2 =$_POST["pick2"];
5. $combo1=5;
6. $combo2=6;
7. if (($pick1 == $combo1) && ($pick2 == $combo2)) {
8.     print ("Congratulations you got both secret numbers
$combo1 $combo2!");
9. } elseif (($pick1 == $combo1) || ($pick2 == $combo2)){
10.    print ("You got one number right.");
11. } else {
12.     print ("Sorry, you are totally wrong!");
13. }
14. print ("You guessed $pick1 and $pick2.");
15. ?></body></html>
```

The Output ...

The previous code can be executed at

<http://webwizard.aw.com/~phppgm/C3/drivelogical.html>

The image displays three sequential screenshots of a web application titled "Number Pick" and "Number Guess Results" using the Netscape 6 browser.

Number Pick - Netscape 6: This window shows two sets of radio buttons for picking numbers from 1 to 9. The first set is labeled "Pick a number from 1 to 9" and the second is labeled "Pick a second number". Below each set of buttons are "Submit Query" and "Reset" buttons. The URL in the address bar is <http://perl-pgm.com/php/C3/drivelogical.html>.

Number Guess Results - Netscape 6: This window displays the results of a guess. It states: "You got one number right. You guessed 5 and 4." The URL in the address bar is <http://65.108.8.8/php/C3/logical.php>.

Number Guess Results - Netscape 6: This window displays the final results. It states: "Congratulations you got both secret numbers 5 6! You guessed 5 and 6." The URL in the address bar is <http://65.108.8.8/php/C3/logical.php>. At the bottom of the page, it says "Document: Done (6.21 secs)".

Question?

