ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Chapter 3.1.
Conditional Statements

Objectives

- Use conditional test statements to compare numerical and string data values
- Use looping statements to repeat statements
- 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 logical operators
  - The if statement above uses the greater than (>) operator to test whether $average is greater than 69.
  - Operators evaluate to *true* or *false*

PHP Test Operators

Operators	Effect	Example	Result

==	Equal to	`if ($x == 6){`
		`    $x = $y + 1;`
		`    $y = $x + 1;`
		`}`
			Run the second and third statements if the value of $x *is equal to* 6.

!=	Not equal to	`if ($x != $y) {`
		`    $x = 5 + 1;`
		`}`
			Run the second statement if the value of $x *is not equal to* the value of $y.

<	Less than	`if ($x < 100) {`
		`    $y = 5;`
		`}`
			Run the second statement if the value of $x *is less than* 100.

>	Greater than	`if ($x > 51) {`
		`    print "OK";`
		`}`
			Run the second statement if the value of $x *is greater than* 51.

>=	Greater than or equal to	`if (16 >= $x) {`
		`    print "x=$x";`
		`}`
			Run the second statement if 16 *is greater than or equal to* the value of $x.

<=	Less than or equal to	`if (16 >= $x) {`
		`    print "x=$x";`
		`}`
			Run the second and third statements if the value of $x *is less than or equal to* the value of $y.

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">
```

Sets $grade1

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

Sets $grade2

Receiving Code

```
<html>
<head><title>Decisions</title></head>
<body>
<?php
    $average = ($grade1 + $grade2) / 2;
    if ( $average > 89 ) {
    print "Average score: $average You got an A! <br>";
    }
    $max=$grade1;
  if ($grade1 < $grade2) {
      $max = $grade2;
    }
  print ("Your max score was $max");
?>
</body></html>
```

Calculate average

Output if $average is more than 89

Set when $grade2 is more than $grade1

Receiving Code With REGISTER_GLOBALS off

```
<html>
<head><title>Decisions</title></head>
<body>
<?php
$grade1= $_POST[“grade1”];
$grade2= $_POST[“grade2”];
    $average = ($grade1 + $grade2) / 2;
    if ( $average > 89 ) {
    print "Average score: $average You got an A! <br>";
    }
    $max=$grade1;
   if ($grade1 < $grade2) {
      $max = $grade2;
    }
   print ("Your max score was $max");
?>
</body></html>
```

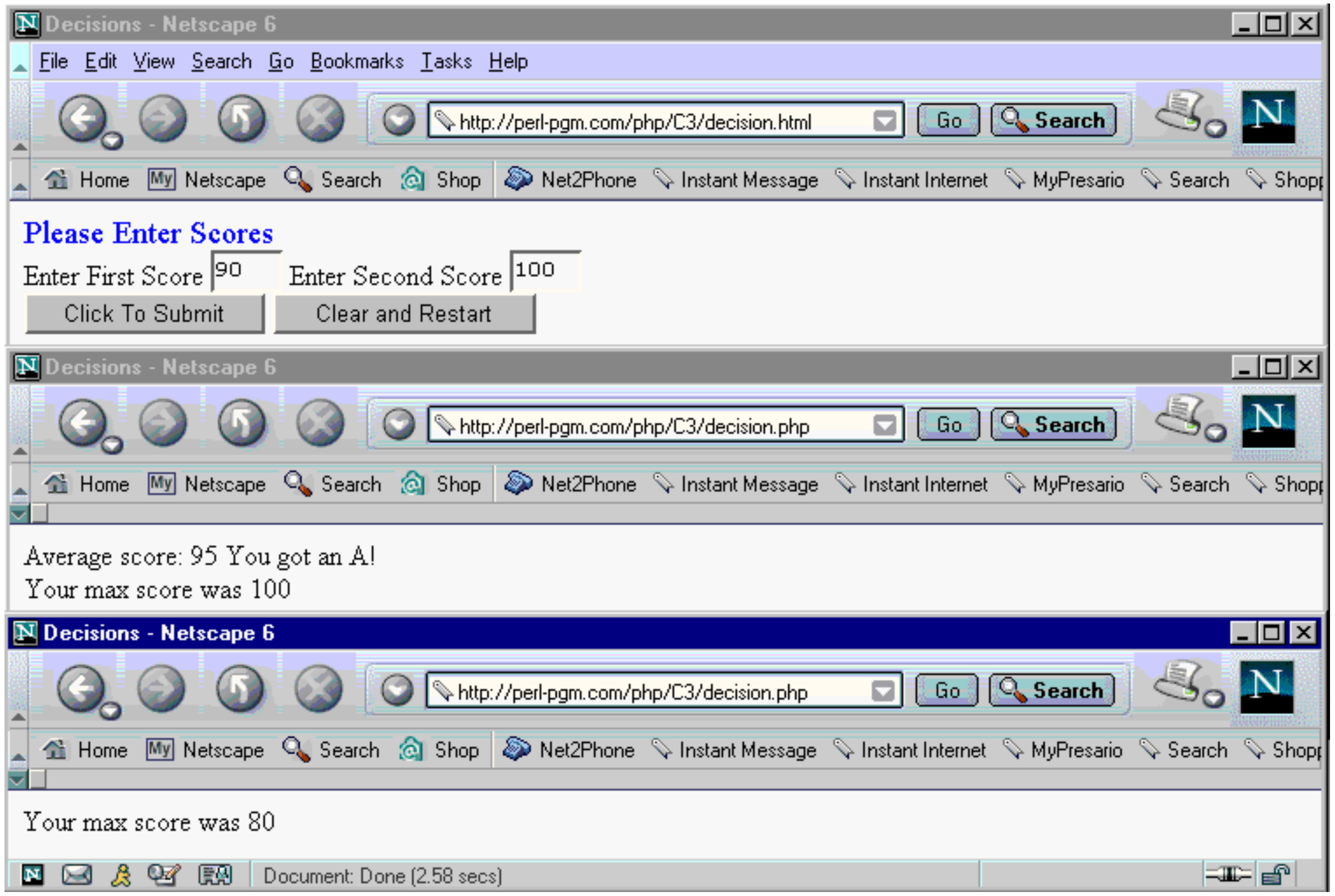
Get grade1 and grade2 from HTML form.

Calculate average

Output if $average is more than 89

Set when $grade2 is more than $grade1

A Full Example ...


Decisions - Netscape 6
File Edit View Search Go Bookmarks Tasks Help
http://perl-pgm.com/php/C3/decision.html
Go
Search
Home
Netscape
Search
Shop
Net2Phone
Instant Message
Instant Internet
MyPresario
Search
Shop

Please Enter Scores

Enter First Score 90 Enter Second Score 100

Click To Submit | Clear and Restart

Decisions - Netscape 6
http://perl-pgm.com/php/C3/decision.php

Average score: 95 You got an A!
Your max score was 100

Decisions - Netscape 6
http://perl-pgm.com/php/C3/decision.php

Your max score was 80

Document: Done (2.58 secs)

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

Sets $first

Sets $second

Receiving Code

```
<html>
<head><title>String Comparison Results</title></head>
<body>
<?php
print ("First=$first Second=$second<br>");
if ($first == $second) {
  print ("$first and $second are equal");
}
if ($first < $second) {
  print ("$first is less than $second");
}
if ($first > $second) {
  print ("$first is greater than $second");
}
?></body></html>
```

Output if $first is equal to $second

Set when $second is less than $first

Set when $first is more than $second

Receiving Code With REGISTER_GLOBALS OFF

```
<html>
<head><title>String Comparison Results</title></head>
<body>
<?php
$first = $_POST[“first”];
$second = $_POST[“second”];
print ("First=$first Second=$second<br>");
if ($first == $second) {
  print ("$first and $second are equal");
}
if ($first < $second) {
  print ("$first is less than $second");
}
if ($first > $second) {
  print ("$first is greater than $second");
}
?></body></html>
```

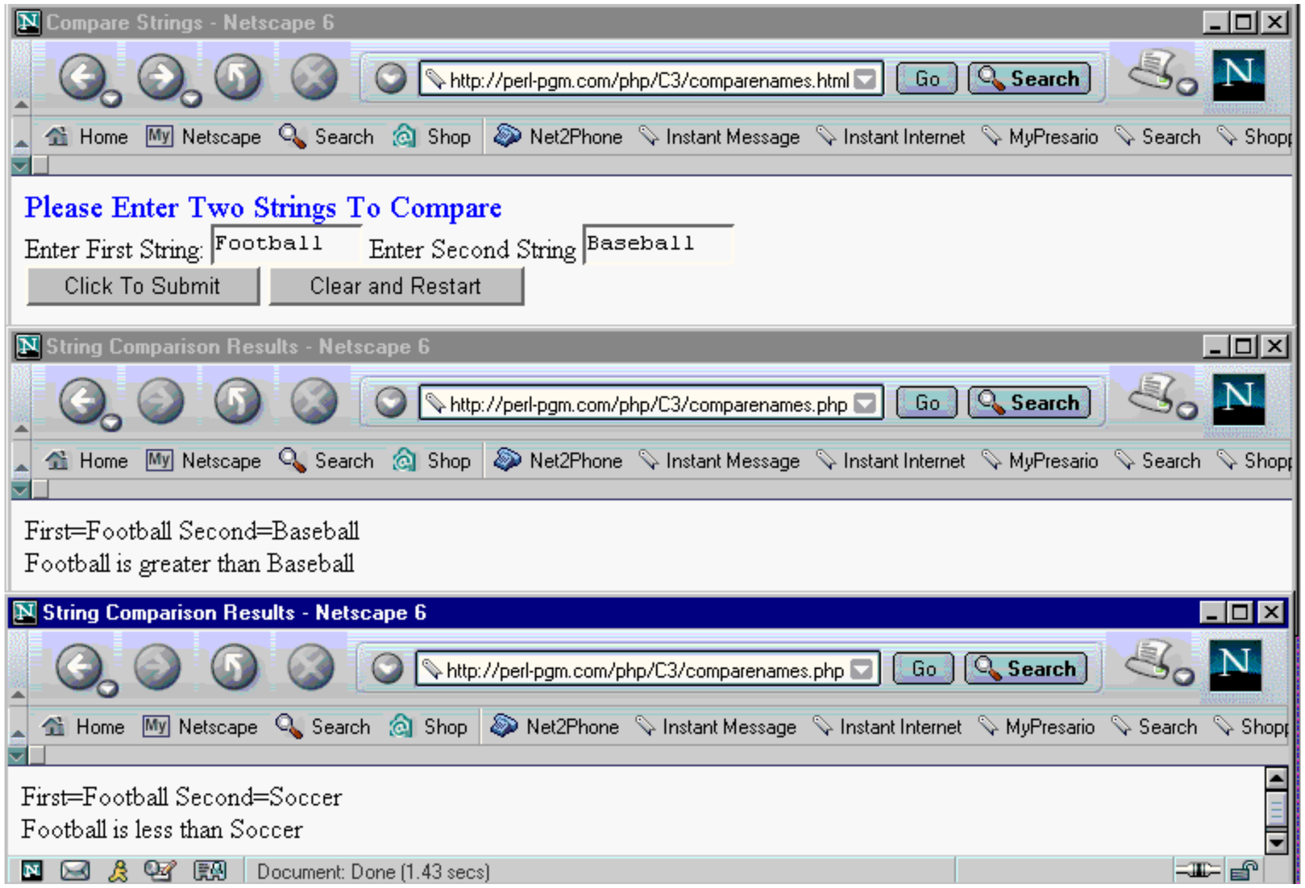
Get the values of $first and $second

Output if $first is equal to $second

Set when $second is less than $first

Set when $first is more than $second

The Output ...


Compare Strings - Netscape 6
http://perl-pgm.com/php/C3/comparenames.html
Go
Search
Home
My Netscape
Search
Shop
Net2Phone
Instant Message
Instant Internet
MyPresario
Search
Shop

Please Enter Two Strings To Compare

Enter First String: Football Enter Second String: Baseball

Click To Submit | Clear and Restart

String Comparison Results - Netscape 6
http://perl-pgm.com/php/C3/comparenames.php

```
First=Football Second=Baseball
Football is greater than Baseball
```

String Comparison Results - Netscape 6
http://perl-pgm.com/php/C3/comparenames.php

```
First=Football Second=Soccer
Football is less than Soccer
```

Document: Done (1.43 secs)

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

Sets $grade1

```
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 $grade2

Receiving Code

```
<html>
<head><title>Grade Calculation</title></head>
<body>
<?php
$average = ($grade1 + $grade2) / 2;
if ($average > 89) {
   print ("Average=$average You got an A");
} elseif ($average > 79) {
   print ("Average=$average You got a B");
} elseif ($average > 69) {
   print ("Average=$average You got a C");
} elseif ($average > 59) {
   print ("Average=$average You got a D");
} elseif ($average >= 0) {
   print ("Grade=$grade You got an F");
} else {
   print ("Illegal average less than 0 average=$average");
}
$max=$grade1;
if ($grade1 < $grade2) {
$max = $grade2;
}
print ("<br>Your max score was $max");
?> </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

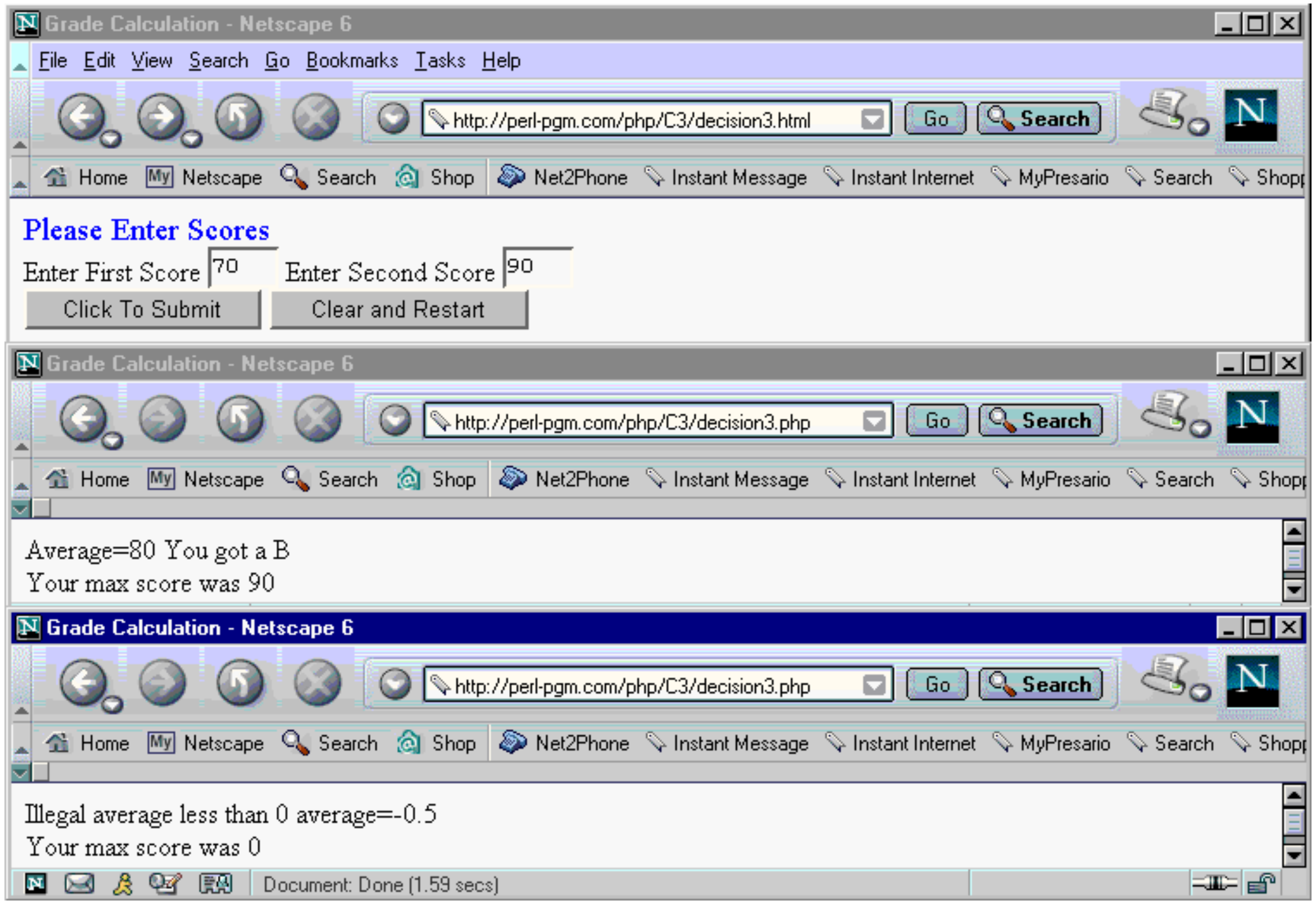
```
<html> <head><title>Grade Calculation</title></head>
<body>
<?php
$grade1 = $_POST[“grade1”]; $grade2 = $_POST[“grade2”];
$average = ($grade1 + $grade2) / 2;
if ($average > 89) {
   print ("Average=$average You got an A");
} elseif ($average > 79) {
   print ("Average=$average You got a B");
} elseif ($average > 69) {
   print ("Average=$average You got a C");
} elseif ($average > 59) {
   print ("Average=$average You got a D");
} elseif ($average >= 0) {
   print ("Grade=$grade You got an F");
} else {
   print ("Illegal average less than 0 average=$average");
}
$max=$grade1;
if ($grade1 < $grade2) {
$max = $grade2;
}
print ("<br>Your max score was $max");
?> </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...


Grade Calculation - Netscape 6
File Edit View Search Go Bookmarks Tasks Help
http://perl-pgm.com/php/C3/decision3.html
Go
Search
Home
Netscape
Search
Shop
Net2Phone
Instant Message
Instant Internet
MyPresario
Search
Shop

Please Enter Scores

Enter First Score 70 Enter Second Score 90

Click To Submit | Clear and Restart

Grade Calculation - Netscape 6
http://perl-pgm.com/php/C3/decision3.php

```
Average=80 You got a B
Your max score was 90
```

Grade Calculation - Netscape 6
http://perl-pgm.com/php/C3/decision3.php

```
Illegal average less than 0 average=-0.5
Your max score was 0
```

Document: Done (1.59 secs)

1.2. Using the switch Statement

- Use switch statement as another conditional test

```
switch ($rating) {
  case 1:
    $rated = "Poor";
    print "The rating was $rated";
    break;
  case 2:
    $rated = "Fair";
    print "The rating was $rated";
    break;
  case 3:
    $rated = "Good";
    print "The rating was $rated";
    break;
  default:
    print "Error: that rating does not exist";
}
```

Enclose in curly brackets

Run these when $rating has value 1.

Run these when $rating has value 2.

Run these when $rating has value 3.

When value not 1, 2, or 3.

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.

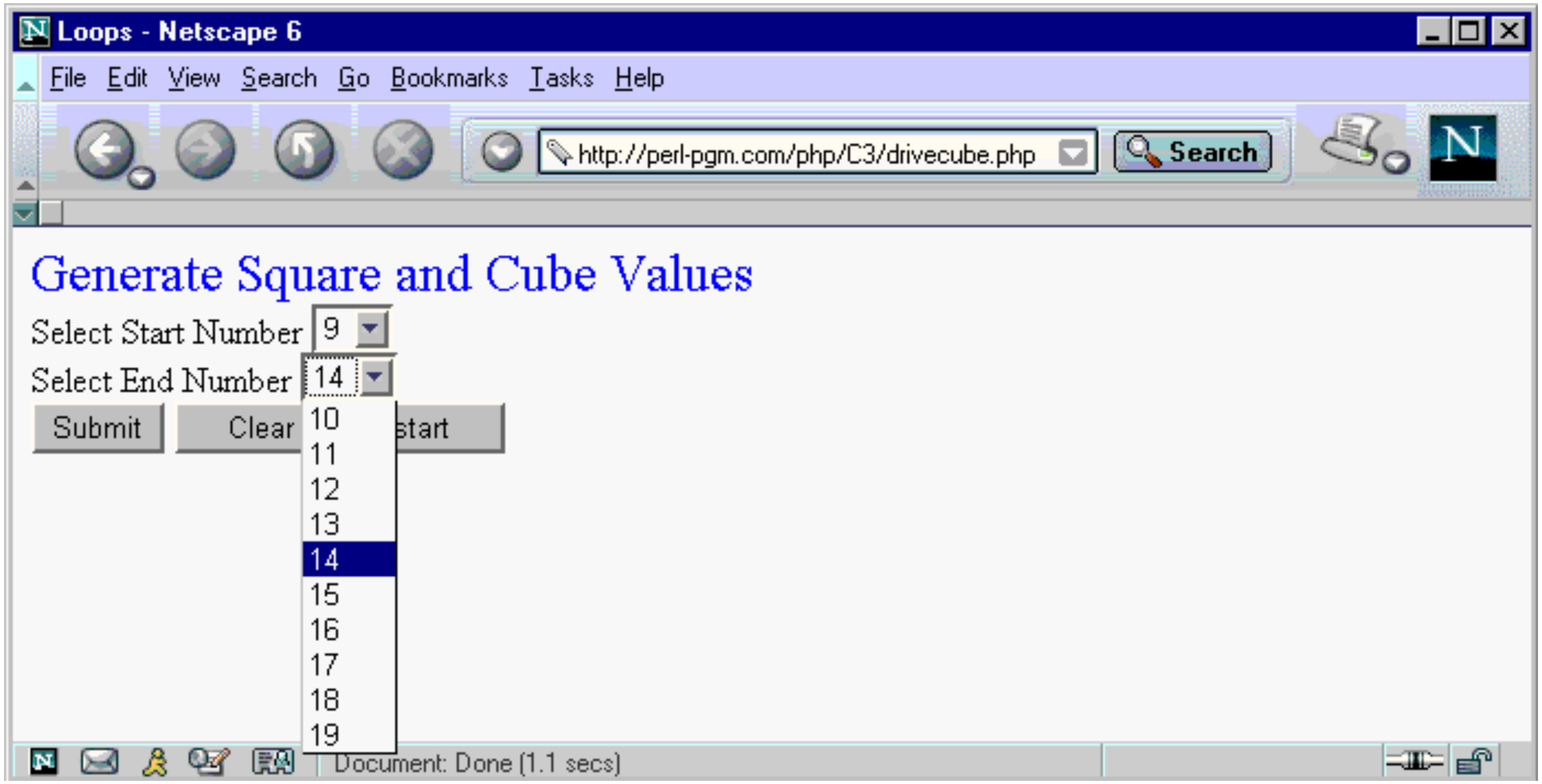
Full Script Example ...

```
<html><head><title>Loops</title></head>
<body><font size="5" color="blue">
Generate Square and Cube Values </font>
<br>
<form action="http://webwizard.aw.com/~phppgm/C3/whileloop.php"
 method="post">
<?php
print ("Select Start Number");
print ("<select name=\"start\">");
for ($i=0; $i<10; $i++) {
      print ("<option>$i</option>");
}
print ("</select>");
print ("<br>Select End Number");
print ("<select name=\"end\">");
for ($i=10; $i<20; $i++) {
     print "(<option>$i</option>)";
}
print ("</select>");
?>
<br><input type="submit" value="Submit">
<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...



Loops - Netscape 6
File Edit View Search Go Bookmarks Tasks Help
http://perl-pgm.com/php/C3/drivecube.php
Search

Generate Square and Cube Values

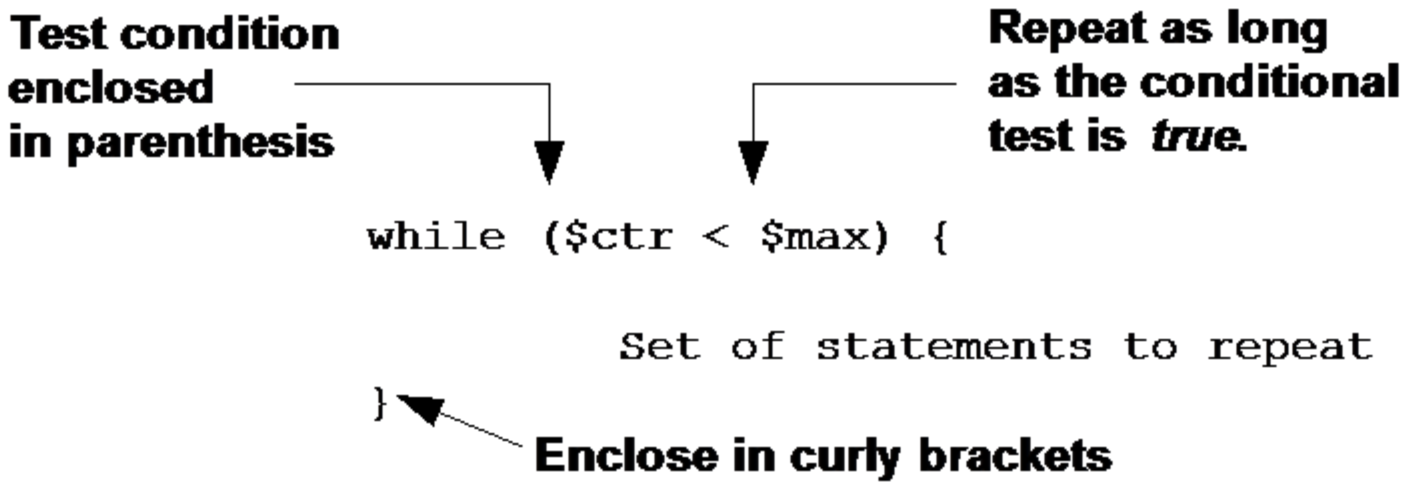
Select Start Number 9

Select End Number 14

10
11
12
13
14
15
16
17
18
19
Submit
Clear
start
Document: Done (1.1 secs)

2.2. Using the while loop

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


Test condition enclosed in parenthesis
Repeat as long as the conditional test is *true*.
Enclose in curly brackets

```
while ($ctr < $max) {

          Set of statements to repeat
}
```

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 ...

```
<html>
<head><title>While Loop</title></head>
<body>
<font size="4" color="blue"> Table of Square and Cube
 Values
</font>
<table border=1>
<th> Numb </th> <th> Sqr </th> <th> Cubed </th>
<?php
   $i = $start;
   while ($i <= $end) {
        $sqr=$i*$i;
        $cubed=$i*$i*$i;
        print
("<tr><td>$i</td><td>$sqr</td><td>$cubed</td></tr>");
        $i = $i + 1;
    }
?></table></body></html>
```

A Full Script Example (with RESGISTER_GLOBALS off)

```
<html>
<head><title>While Loop</title></head>
<body>
<font size="4" color="blue"> Table of Square and Cube
  Values </font>
<table border=1>
<th> Numb </th> <th> Sqr </th> <th> Cubed </th>
<?php
   $start = $_POST[“start”];  $end = $_POST[“end”];
   $i = $start;
    while ($i <= $end) {
       $sqr=$i*$i;
       $cubed=$i*$i*$i;
      print
  ("<tr><td>$i</td><td>$sqr</td><td>$cubed</td></tr>");
       $i = $i + 1;
    }
?></table></body></html>
```

The Output ...

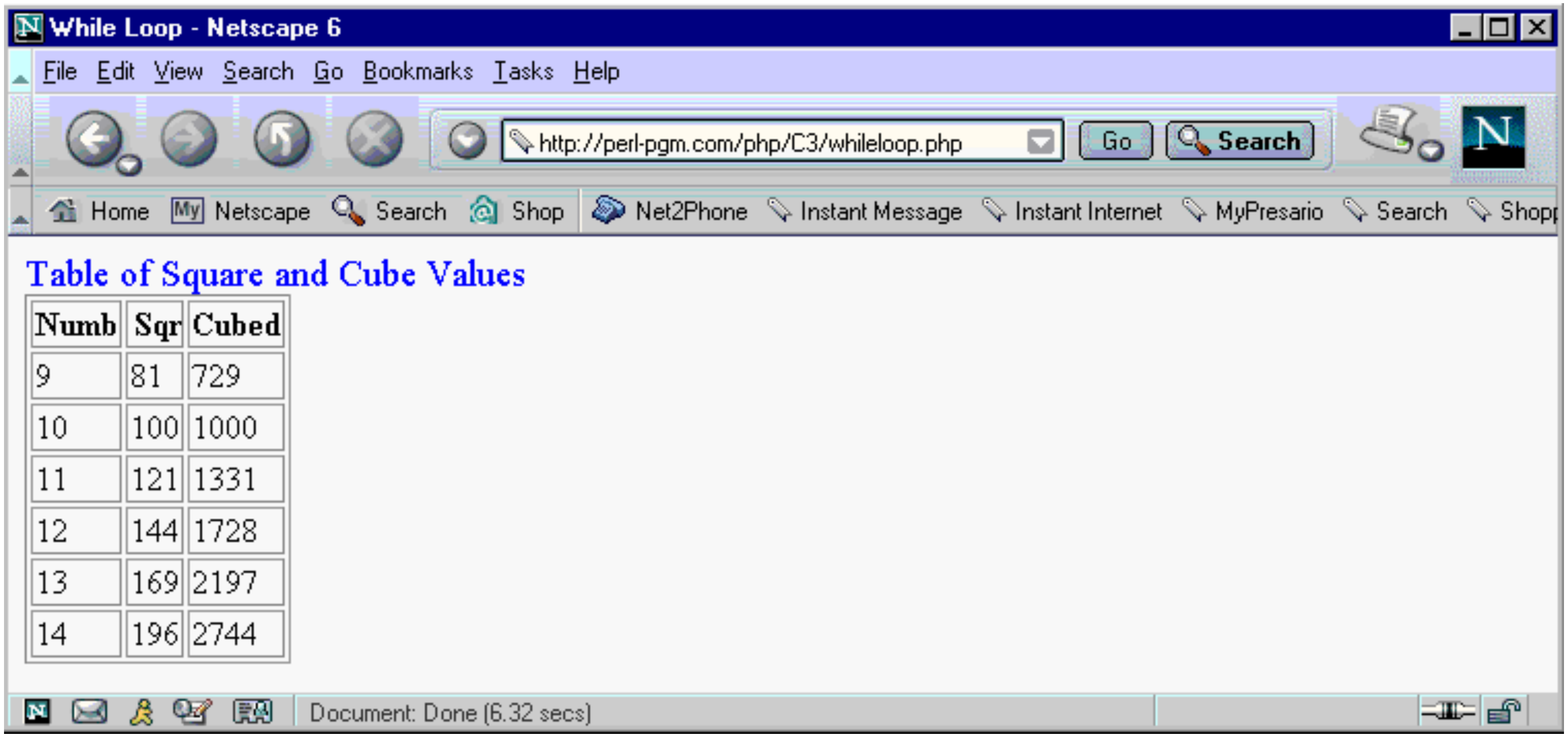


Table of Square and Cube Values

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

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

- *U*sed much like the AND operator in if statements andwhile 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

- *U*sed to test whether an expression is *fals*e (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.
    - <font size=4 > Pick a number from 1 to 9 <br>
    - <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

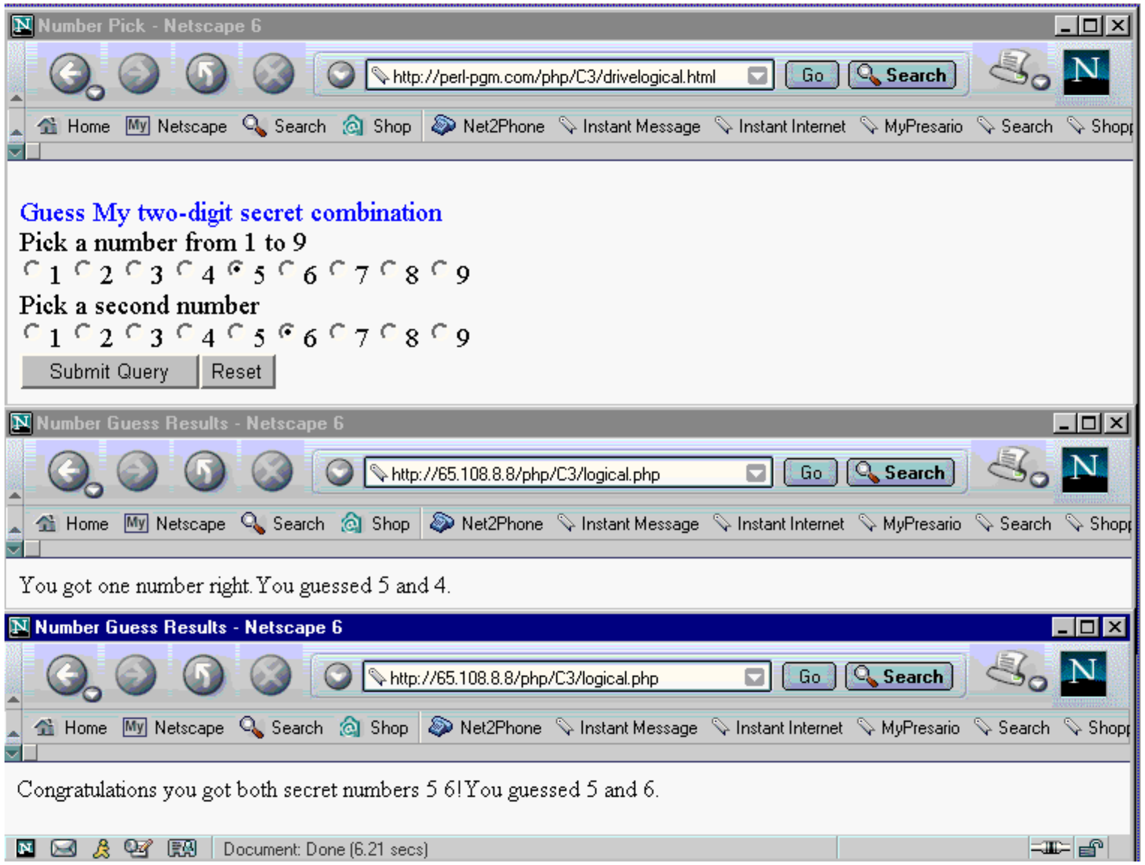
A Full Script Example ...

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

A Full Script Example … with REGISTER_GLOBALS off

```
<html><head><title>Number Guess Results </title><head>
<body>
<?php
$pick1 =$_POST[“pick1”]; $pick2 =$_POST[“pick2”];
$combo1=5;
$combo2=6;
if (($pick1 == $combo1) && ($pick2 == $combo2)) {
     print ("Congratulations you got both secret numbers $combo1 $combo2!");
} elseif (($pick1 == $combo1) || ($pick2 == $combo2)){
     print ("You got one number right.");
} else {
     print ("Sorry, you are totally wrong!");
}
print ("You guessed $pick1 and $pick2.");
?></body></html>
```

The Output ...


Number Pick - Netscape 6
http://perl-pgm.com/php/C3/drivelogical.html
Go
Search
Home
Netscape
Search
Shop
Net2Phone
Instant Message
Instant Internet
MyPresario
Search

Guess My two-digit secret combination

Pick a number from 1 to 9

1 2 3 4 [✓ 5] 6 7 8 9

Pick a second number

1 2 3 4 5 [✓ 6] 7 8 9

Submit Query Reset

Number Guess Results - Netscape 6
http://65.108.8.8/php/C3/logical.php

You got one number right. You guessed 5 and 4.

Number Guess Results - Netscape 6
http://65.108.8.8/php/C3/logical.php

Congratulations you got both secret numbers 5 6! You guessed 5 and 6.

Document: Done (6.21 secs)

Summary

- Use conditional statements to test for certain conditions and, based on the results of the test, to run specific script statements.
- Loops expand the types of programming problems that you can solve and allow you to solve some programming problems much more concisely
- Use logical AND (&&), OR (||) and NOT (!) operators to carry out compound tests.

Summary

- Variables are used to store and access data in computer memory. You can associate a value with a variable, change that value, print it out, and perform many different operations on it.
- PHP supports both numeric and string variables. String variables use different methods for value manipulation (for example, concatenation) than numeric variables do.

Question?