CIS 4004: Web-Based Information Technology
Spring 2011

Introduction to PHP – Part 1

Instructor : Dr. Mark Llewellyn
markl@cs.ucf.edu
HEC 236, 407-823-2790
http://www.cs.ucf.edu/courses/cis4004/spr2011

Department of Electrical Engineering and Computer Science
University of Central Florida

Introduction to PHP

- PHP was created by Rasmus Lerdorf in 1994, He first used it to build extensions into HTML documents to enhance his personal home page.

- In fact, PHP was originally called Personal Home Page. As Lerdorf freely distributed the program source, PHP gained popularity and became an Apache Software Foundation project. Eventually, PHP's name was changed to PHP Hypertext Preprocessor.

- In late 2010 there were slightly more than 1 trillion websites registered globally. PHP estimates that more than 230 million websites are PHP enabled to some extent.

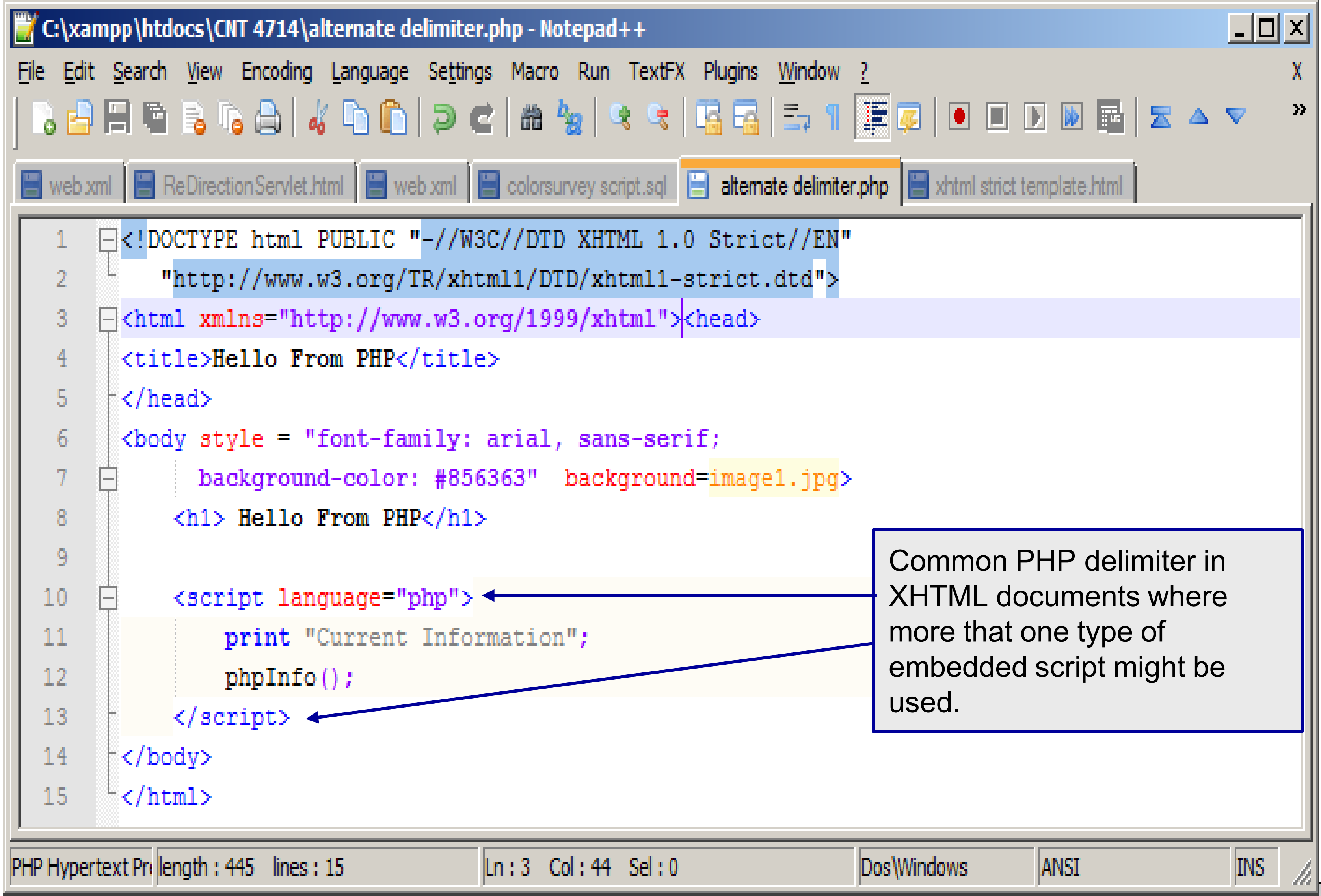
Introduction to PHP

- PHP scripts can be created with any text editor, although Notepad++ is quite convenient for PHP scripting. I'll primarily use it in the examples. NetBeans also provides a fairly decent environment for PHP development including a debugger.

- PHP script files should be saved with a `.php` extension.

- When PHP is embedded inside XHTML documents, as it commonly is, several different delimiters can be used. These are illustrated on the next page.

Introduction to PHP



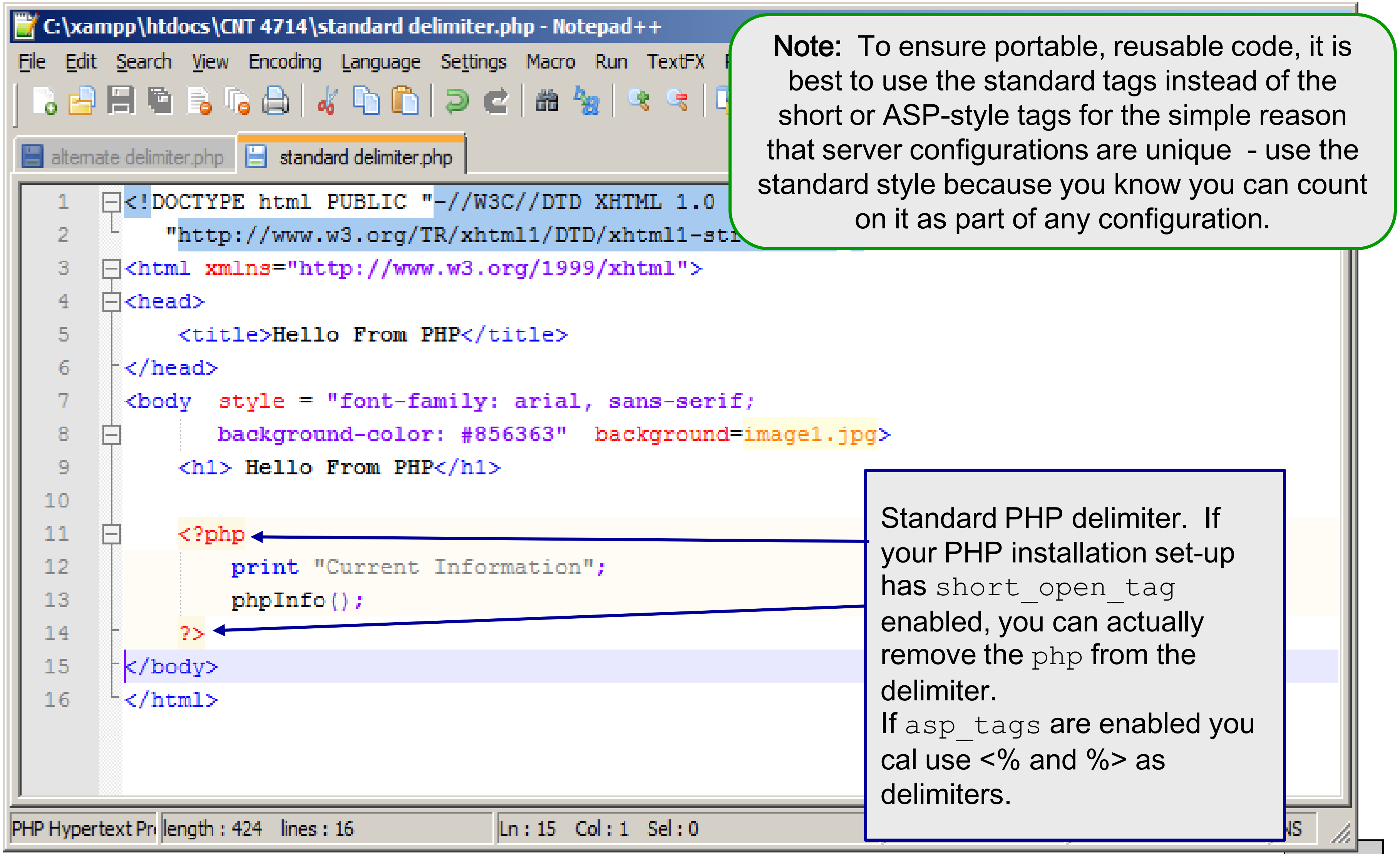
C:\xampp\htdocs\CNT 4714\alternate delimiter.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
web.xml
ReDirectionServlet.html
web.xml
colorsurvey script.sql
alternate delimiter.php
xhtml strict template.html
PHP Hypertext Pr length : 445 lines : 15
Ln : 3 Col : 44 Sel : 0
Dos\Windows
ANSI
INS

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><head>
<title>Hello From PHP</title>
</head>
<body style = "font-family: arial, sans-serif;
      background-color: #856363"  background=image1.jpg>
    <h1> Hello From PHP</h1>

    <script language="php">
       print "Current Information";
       phpInfo();
    </script>
</body>
</html>
```

Common PHP delimiter in XHTML documents where more that one type of embedded script might be used.

Introduction to PHP


C:\xampp\htdocs\CNT 4714\standard delimiter.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX
alternate delimiter.php
standard delimiter.php
PHP Hypertext Pr
length : 424 lines : 16
Ln : 15 Col : 1 Sel : 0
NS

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-str
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Hello From PHP</title>
</head>
<body  style = "font-family: arial, sans-serif;
       background-color: #856363"  background=image1.jpg>
    <h1> Hello From PHP</h1>

    <?php
        print "Current Information";
        phpInfo();
    ?>
</body>
</html>
```

Note: To ensure portable, reusable code, it is best to use the standard tags instead of the short or ASP-style tags for the simple reason that server configurations are unique - use the standard style because you know you can count on it as part of any configuration.

Standard PHP delimiter. If your PHP installation set-up has `short_open_tag` enabled, you can actually remove the `php` from the delimiter.
If `asp_tags` are enabled you cal use <% and %> as delimiters.

Introduction to PHP

- As with any programming language, good practice in writing scripts would require comments to be included within the script.
- In-line comments in PHP are indicated with two forward slashes (//).
- Comments can appear any where in the script file and can appear in any position on any line.
- Multiple line comments are delimited with /* and */
- Most PHP implementations also allow # to delimit in-line comments.

Variables In PHP

- You can select just about any set of characters for a variable name in PHP, but they must:
  - Use a dollar sign ($) as the first character.
  - Use a letter or an underscore character (_) as the second character.
- As with any programming/scripting language, good practice would suggest selecting variable names that help describe their function. For example `$counter` is more descriptive than `$c` or `$ctr`.
- You can use the `echo` statement or the `print()` function to output data in PHP. Which you use is more a matter of personal taste or style than anything else.

Variables In PHP

- To print out the value of a variable $x, write the following PHP statement:

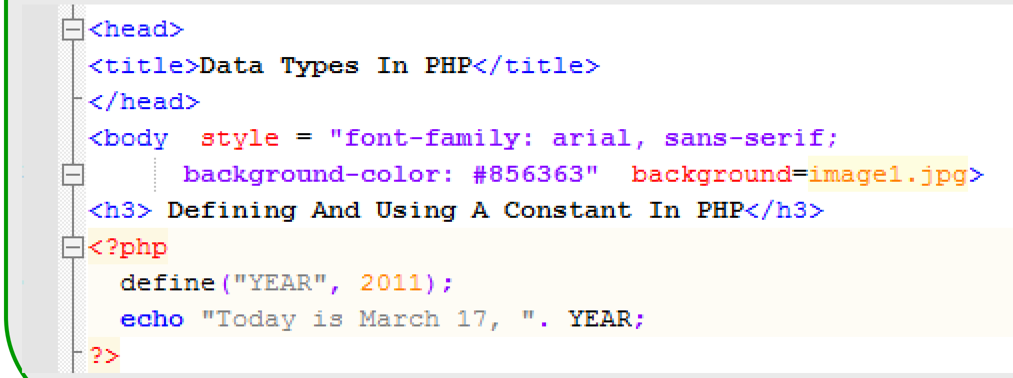
```
print ("$x");
```

- The following code will output “Candice is 26 years old”.

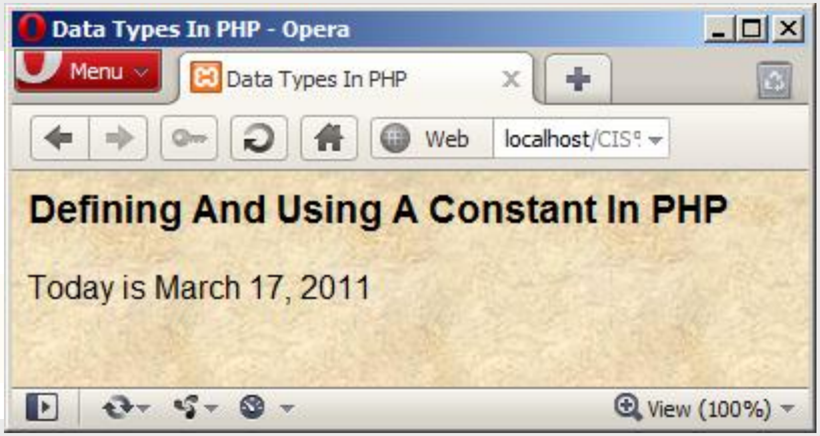
```
$age=26;
print (“Candice is $age years old.");
```

- The next page illustrates a full example using PHP variables.

Note: Constants are defined in PHP using the built-in `define()` function. As its name would imply a constant’s value cannot be changed once it is set.



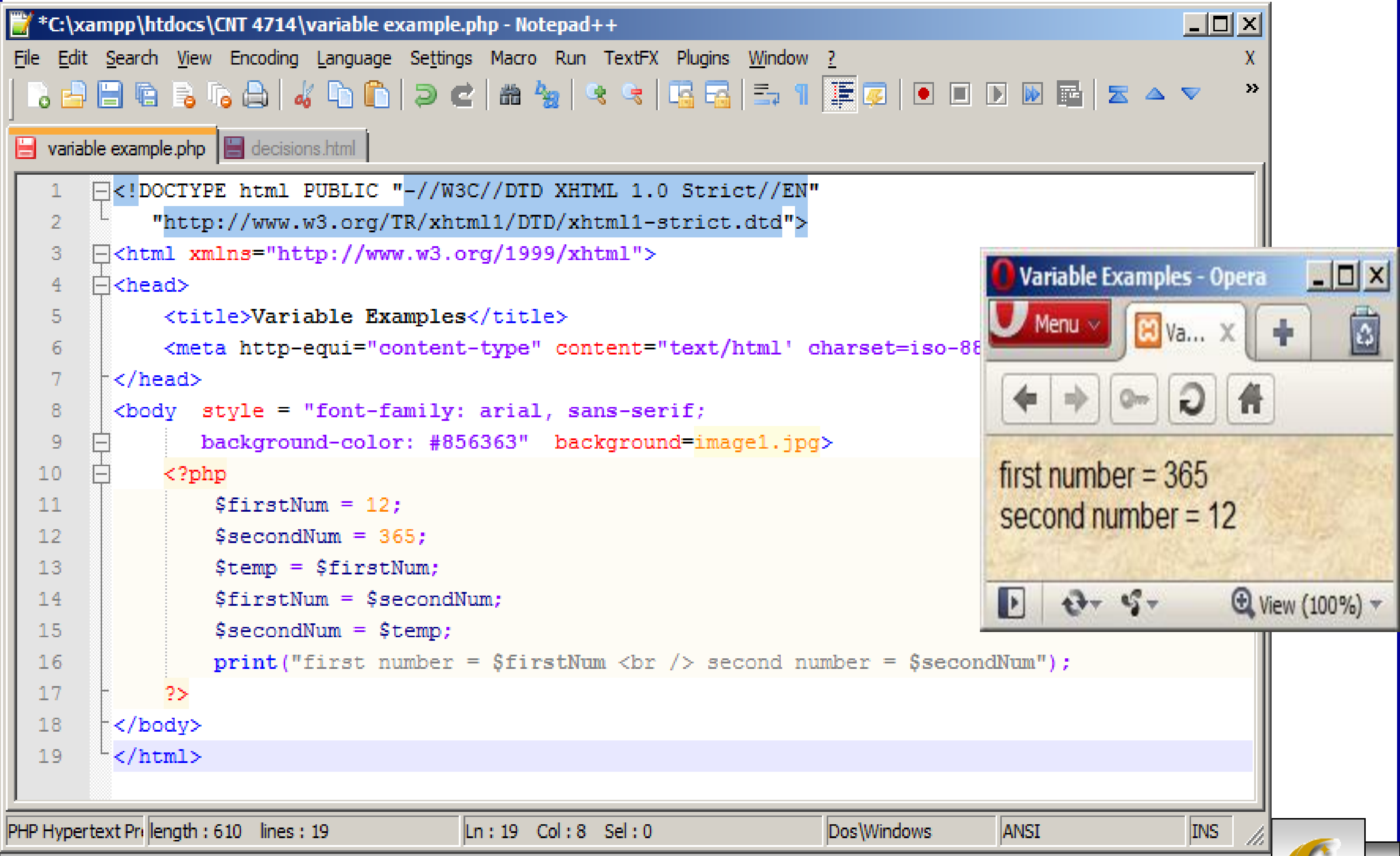
```
<head>
<title>Data Types In PHP</title>
</head>
<body  style = "font-family: arial, sans-serif;
        background-color: #856363"  background=image1.jpg>
<h3> Defining And Using A Constant In PHP</h3>
<?php
  define("YEAR", 2011);
  echo "Today is March 17, ". YEAR;
?>
```



Defining And Using A Constant In PHP

Today is March 17, 2011

Variables In PHP


*C:\xampp\htdocs\CNT 4714\variable example.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
variable example.php
decisions.html
PHP Hypertext Pr
length : 610 lines : 19
Ln : 19 Col : 8 Sel : 0
Dos\Windows
ANSI
INS
Variable Examples - Opera
Menu
Va...
View (100%)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Variable Examples</title>
    <meta http-equi="content-type" content="text/html' charset=iso-88
</head>
<body  style = "font-family: arial, sans-serif;
       background-color: #856363"  background=image1.jpg>
    <?php
        $firstNum = 12;
        $secondNum = 365;
        $temp = $firstNum;
        $firstNum = $secondNum;
        $secondNum = $temp;
        print("first number = $firstNum <br /> second number = $secondNum");
    ?>
</body>
</html>
```

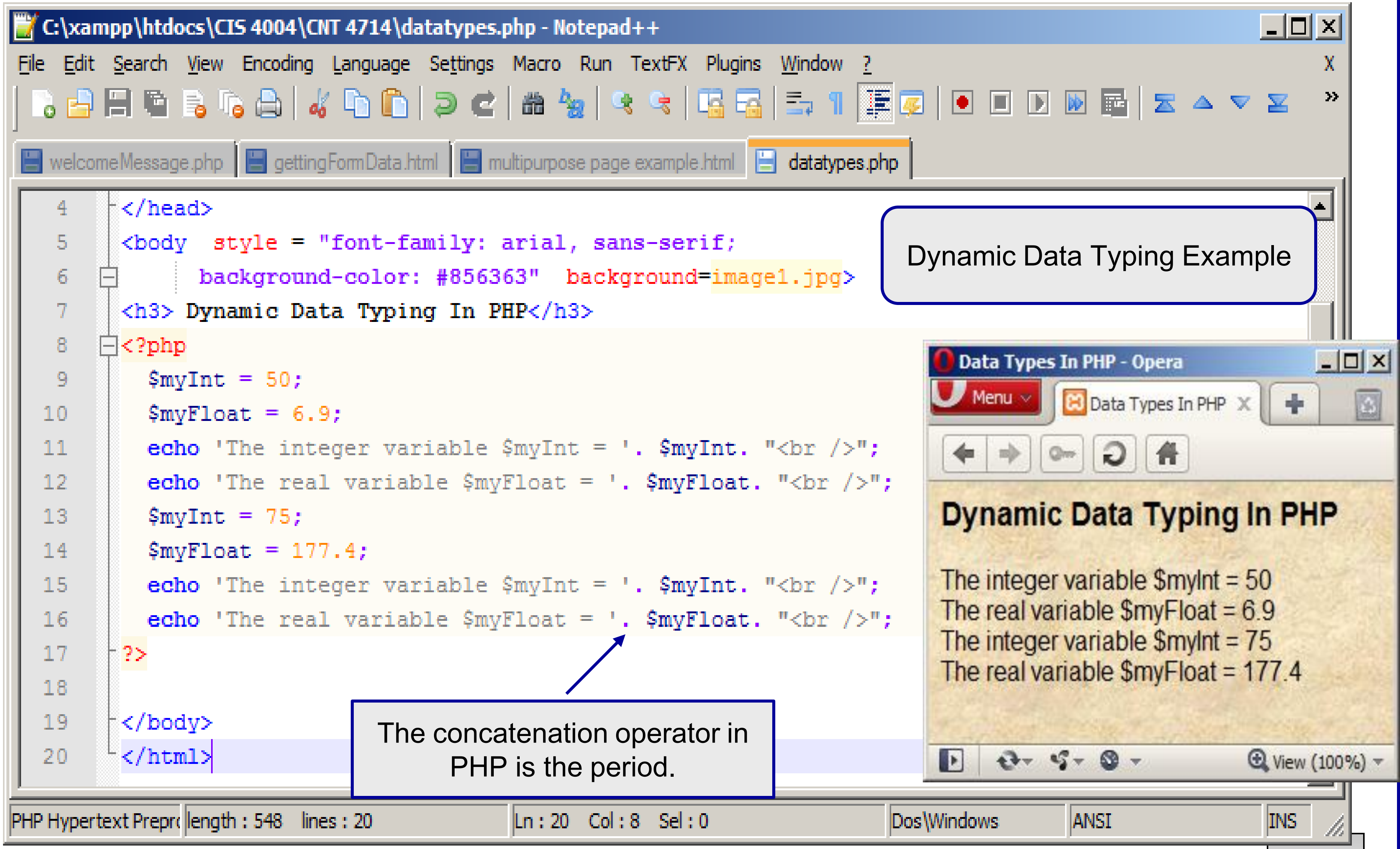
```
first number = 365
second number = 12
```

Data Types In PHP

- PHP is a dynamically typed language. This basically means that variables are not assigned a type when the variable is declared. Variable type is determined through assignment.
- The standard data types in PHP are shown in the table below:

Data Type	Example	Description
Boolean	true	Either `true` or `false`
Integer	5	A whole number
Float or Double	3.14159	A floating-point number
String	“Hello”	A collection of characters
Object		An instance of a class
Array		An ordered set of keys and values
Resource		Reference to a 3rd party resource (e.g. a database)
NULL		An uninitialized variable

Data Types In PHP


C:\xampp\htdocs\CIS 4004\CNT 4714\datatypes.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
welcomeMessage.php
gettingFormData.html
multipurpose page example.html
datatypes.php
PHP Hypertext Prepr length : 548 lines : 20
Ln : 20 Col : 8 Sel : 0
Dos\Windows
ANSI
INS
Data Types In PHP - Opera
Menu
Data Types In PHP
View (100%)

```
</head>
<body  style = "font-family: arial, sans-serif;
       background-color: #856363"  background=image1.jpg>
<h3> Dynamic Data Typing In PHP</h3>
<?php
  $myInt = 50;
  $myFloat = 6.9;
  echo 'The integer variable $myInt = '. $myInt. "<br />";
  echo 'The real variable $myFloat = '. $myFloat. "<br />";
  $myInt = 75;
  $myFloat = 177.4;
  echo 'The integer variable $myInt = '. $myInt. "<br />";
  echo 'The real variable $myFloat = '. $myFloat. "<br />";
?>

</body>
</html>
```

Dynamic Data Typing Example

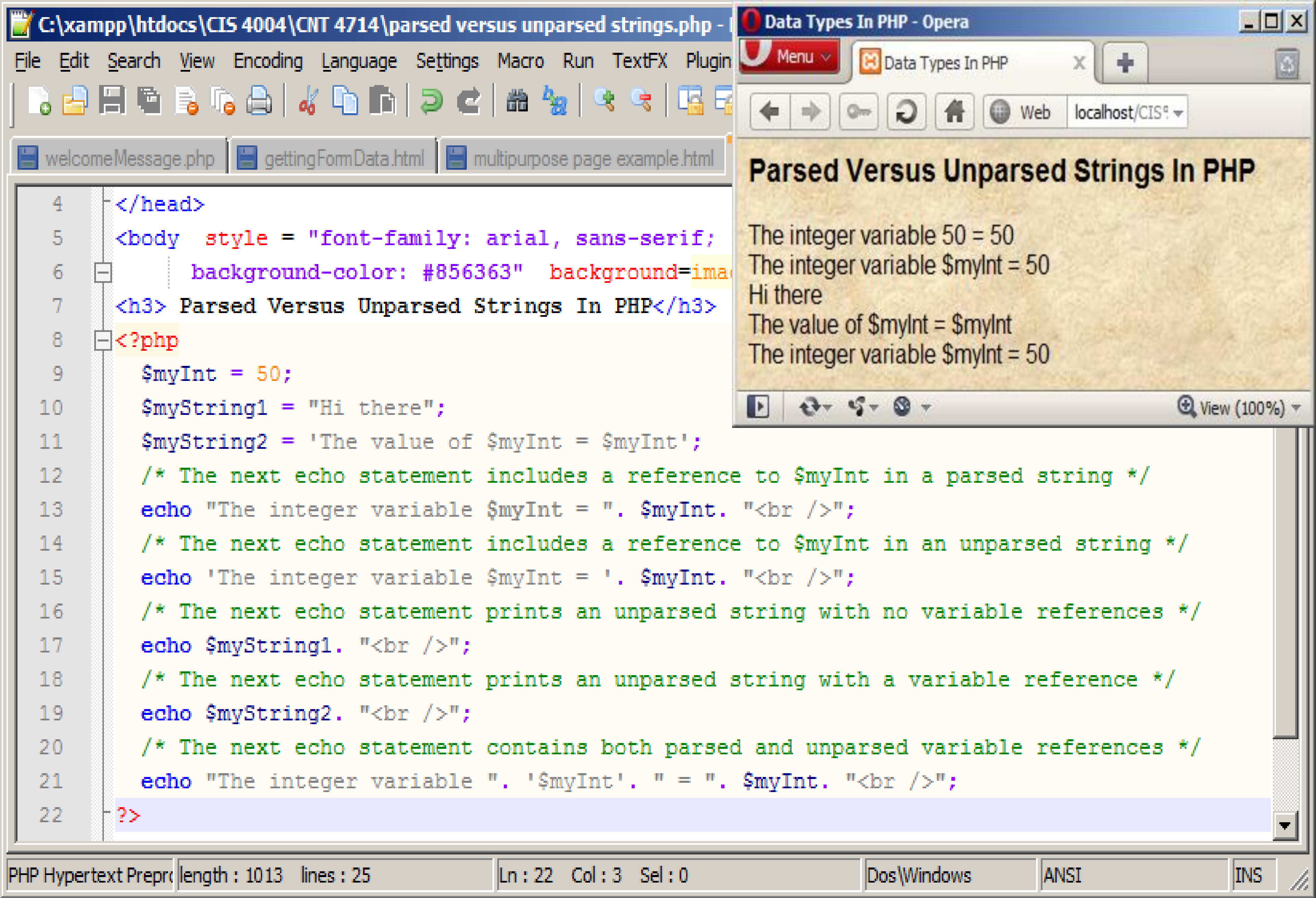
Dynamic Data Typing In PHP

```
The integer variable $myInt = 50
The real variable $myFloat = 6.9
The integer variable $myInt = 75
The real variable $myFloat = 177.4
```

The concatenation operator in PHP is the period.

Data Types In PHP

- Technically speaking, there are two types of strings in PHP: parsed and unparsed.
- Parsed strings are defined using double quotes and are parsed by PHP.
- Unparsed strings are defined using single quotes and are taken as is (they are not parsed).
- What’s the difference? Within a parsed string, any references to variables within that string will be automatically replaced with their respective values, whereas within an unparsed string nothing is replaced.
- The example on the next page will clarify the differences.


C:\xampp\htdocs\CIS 4004\CNT 4714\parsed versus unparsed strings.php -
File Edit Search View Encoding Language Settings Macro Run TextFX Plugin
welcomeMessage.php
gettingFormData.html
multipurpose page example.html
Data Types In PHP - Opera
Menu
Data Types In PHP
Web
localhost/CIS
View (100%)
PHP Hypertext Prepro
length : 1013 lines : 25
Ln : 22 Col : 3 Sel : 0
Dos\Windows
ANSI
INS

```
</head>
<body  style = "font-family: arial, sans-serif;
        background-color: #856363"  background=ima
<h3> Parsed Versus Unparsed Strings In PHP</h3>
<?php
  $myInt = 50;
  $myString1 = "Hi there";
  $myString2 = 'The value of $myInt = $myInt';
  /* The next echo statement includes a reference to $myInt in a parsed string */
  echo "The integer variable $myInt = ". $myInt. "<br />";
  /* The next echo statement includes a reference to $myInt in an unparsed string */
  echo 'The integer variable $myInt = '. $myInt. "<br />";
  /* The next echo statement prints an unparsed string with no variable references */
  echo $myString1. "<br />";
  /* The next echo statement prints an unparsed string with a variable reference */
  echo $myString2. "<br />";
  /* The next echo statement contains both parsed and unparsed variable references */
  echo "The integer variable ". '$myInt'. " = ". $myInt. "<br />";
?>
```

Parsed Versus Unparsed Strings In PHP

```
The integer variable 50 = 50
The integer variable $myInt = 50
Hi there
The value of $myInt = $myInt
The integer variable $myInt = 50
```

Arithmetic Operations In PHP

- PHP supports all normal arithmetic operators, with the normal semantic associated with each operator.

Operator	Effect	Example	Result
`+`	Addition	`$x = 2 + 2;`	`$x` is assigned 4.

`-`	Subtraction	`$y = 3;`
		`$y = $y - 1;`
			`$y` is assigned 2.

`/`	Division	`$y = 14 / 2;`	`$y` is assigned 7.

`*`	Multiplication	`$z = 4;`
		`$y = $z * 4;`
			`$y` is assigned 16.

`%`	Remainder	`$y = 14 % 3;`	`$y` is assigned 2.

- PHP supports automatic increment and decrement operations in both prefix and postfix form, i.e., -- and ++.
- Using an unassigned variable in an expression does not generate an error, the value is simply assumed to be null.

```
<?php
  $y = 3;
  $y = $y + $x + 1;
  print(“x=$x y=$y”);
?>
```

The output is: `x=y=4`

String Variables In PHP

- PHP supports character string variables and this is a widely used aspect of PHP in handling form data.
- Be careful in PHP not to mix numeric and string types together in an expression.
- For example, you might expect the following statements to generate an error message, but they will not. Instead, they will output “y=1”.

```
<?php
   $x = “banana”;
   $sum = 1 + $x’
   print(“y=$sum”);
?>
```

String Variables In PHP

- The string concatenation operator in PHP is the period as shown below:

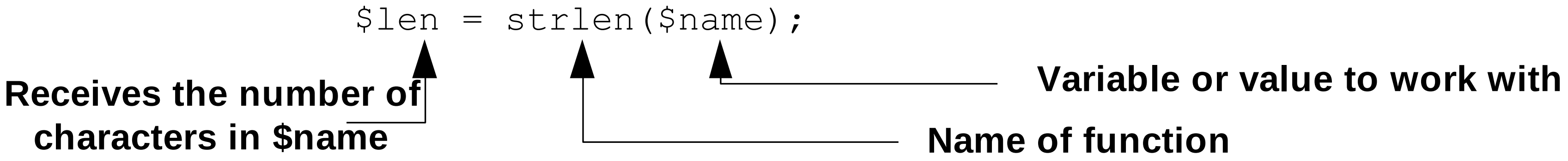
```
<?php
    $firstname = “Megan”;
    $lastname = “Fox”
    $fullname = $firstname . $lastname;
    print(“Full name = $fullname”);
?>
```

The output of this script would be: `Fullname=MeganFox`

You can also use double quotation marks to create concatenation directly. Using the above example you could do the following: $fullname2 = “$firstname $lastname”; This would have the same effect as: $fullname2 = $firstname . $lastname;

String Variables In PHP

- PHP supports a large variety of string handling functions. A few of the more commonly used ones are illustrated on the next few pages.
- Most string functions require you to send them one or more arguments.
- Arguments are input values that functions use in the processing they do.
- Often functions return a value to the script based on the input arguments. For example:



```
$len = strlen($name);
```

Receives the number of characters in $name

Name of function

Variable or value to work with

String Variables In PHP

`strlen()` function:

- This function returns the number of characters in the string argument to the function. Consider the following script:

```
<?php
        $comments = "Good Job";
        $len = strlen($comments);
        print ("Length=$len");
  ?>
```

This PHP script would output “`Length=8`”.

String Variables In PHP

`trim()` function:

- This function removes any blank characters from the beginning and end of a string. For example, consider the following script:

```
<?php
  $in_name = ‟      Megan  Fox  ";
  $name = trim($in_name);
  print ("name=$name$name");
?>
```

This PHP script would output “`name=Megan  FoxMegan  Fox`”.

String Variables In PHP

strtolower() and strtoupper functions:

- These functions return the argument string in all uppercase or all lowercase letters, respectively. For example, consider the following script:

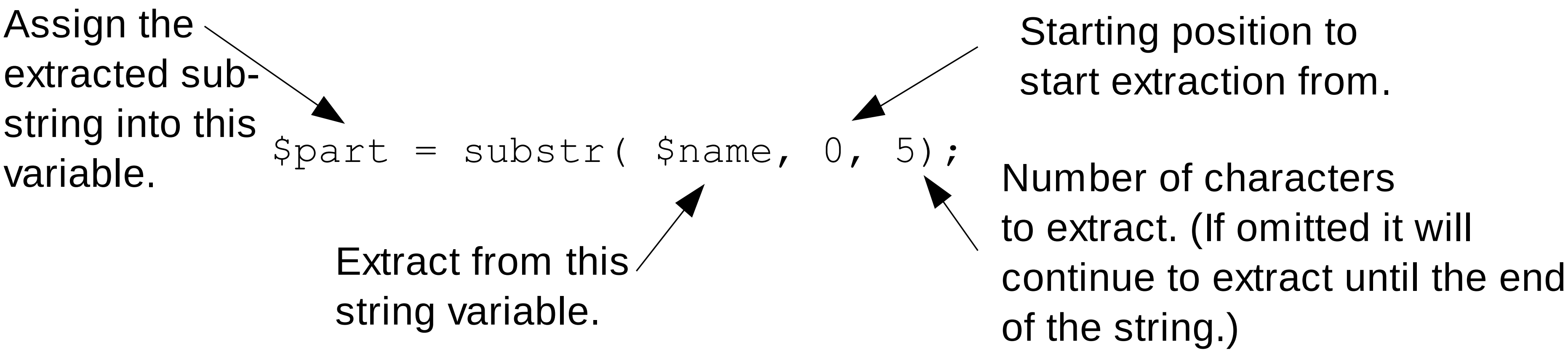
```
<?php
   $inquote = “Now Is The Time";
   $lower = strtolower($inquote);
   $upper = strtoupper($inquote);
   print(“upper=$upper lower=$lower”);
?>
```

This PHP script would output “upper=NOW IS THE TIME lower = now is the time”

String Variables In PHP

`substr()` function:

- This function enables a PHP script to extract a portion of the characters in a string variable. The general syntax is:



```
$part = substr( $name, 0, 5);
```

Assign the extracted sub-string into this variable.

Starting position to start extraction from.

Extract from this string variable.

Number of characters to extract. (If omitted it will continue to extract until the end of the string.)

String Variables In PHP

substr() function:

- The substr() function enumerates character positions starting with 0 (not 1),
  - For example, in the string “Homer”, the “H” would be position 0, the “o” would be position 1, the “m” position 2, and so on.
- For example, the following would output “Month=12 Day=25”.

```
<?php
  $date = "12/25/2002";
  $month = substr($date, 0, 2);
  $day = substr($date, 3, 2);
  print ("Month=$month Day=$day");
?>
```

String Variables In PHP

`substr()` function:

- This example does not include the third argument (and thus returns a substring from the starting position to the end of the search string).

```
<?php
  $date = "12/25/2010";
  $year = substr($date, 6);
  print ("Year=$year");
?>
```

- The above script segment would output `“Year=2010”.`

Controlling Script Flow In PHP

- PHP contains the normal control statements that handle decision making and iteration within a script.
- Normal logical operators are all supported with their standard semantic.
- As with many modern programming and scripting languages remember to use == in a logical comparison operation and not =. The single equal sign is an assignment operator and as such is always true. No syntax error is generated.
- The table on the following page illustrates the common logical operators in PHP.

Note: PHP also contains a === logical comparison operator (called the identical operation). This binary operator returns true iff its two operands are equal in value and also have the same type.

Controlling Script Flow In PHP

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

Controlling Script Flow In PHP

- PHP includes several different forms of logical control statements (decision statements).
- The `if` statement has the form:

```
if (expression) {
        //code to execute if expression evaluates to true
}
```

- The `if-else` statement has the form:

```
if (expression) {
        //code to execute if expression evaluates to true
} else {
        //code to execute when expression evaluates to false
}
```

Controlling Script Flow In PHP

- There is also an `elseif` clause that can be used with `if` statements for a nested stack of `if` statements. The basic syntax for this clause is:

```
if (expression) {
        //code to execute if expression evaluates to true
} elseif (another expression) {
        //code to execute when expression evaluates to false
        //and another expression evaluates to true
} else {
        //code to execute if all expressions evaluate to false
}
```

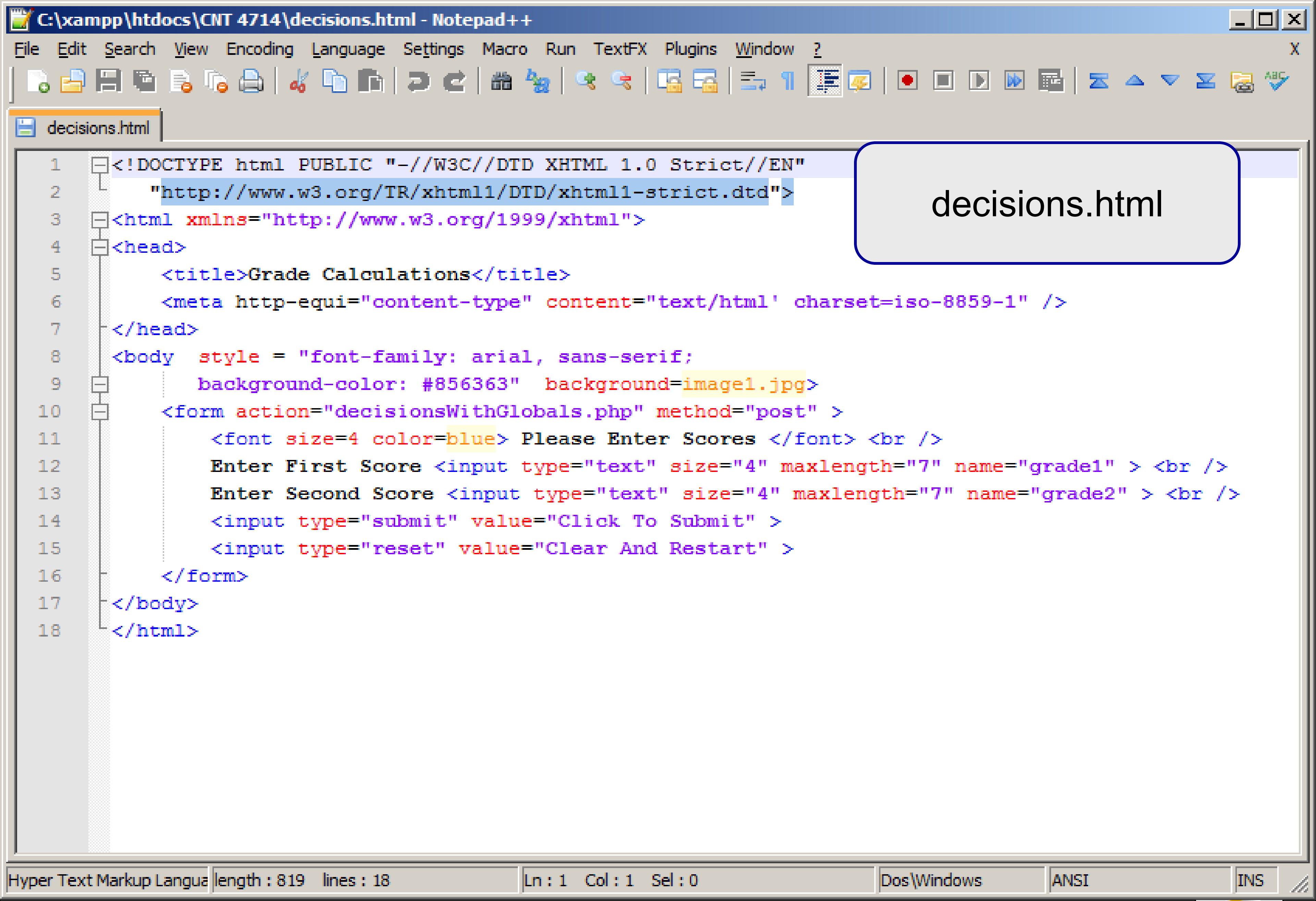
Controlling Script Flow In PHP

- PHP includes a `switch` statement which allows for multiple options for a single evaluation of an expression. The basic syntax for the `switch` statement is:

```
switch (expression) {
     case result1:
            //code to execute if expression evaluates to result1
            break;
     case result2:
            //code to execute if expression evaluates to result2
            break;
     . . .
     default:
            //code to execute if no break has been encountered
 }
```

Controlling Script Flow In PHP

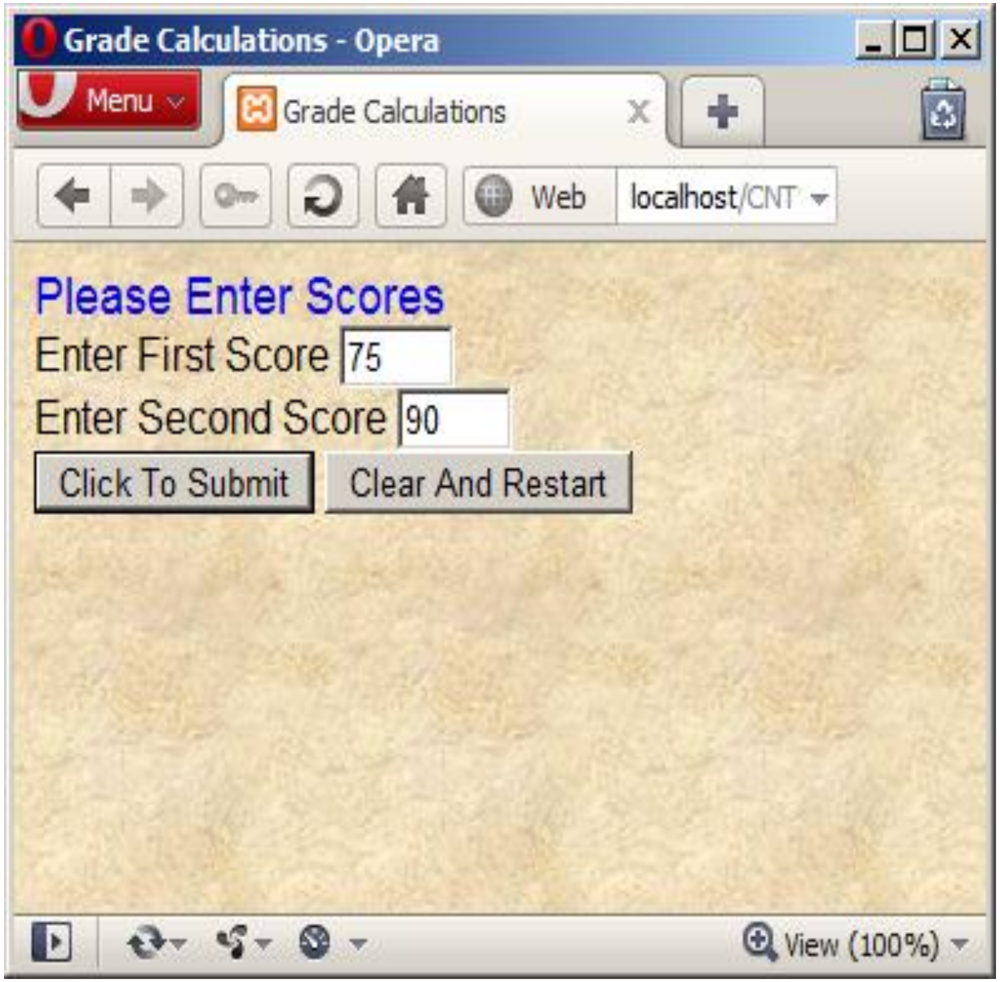
- The following example uses an input form (XHTML) and two values are extracted from the form (`grade1` and `grade2`), passed to a PHP script which determines the average score, the maximum score and assigns a grade to the average for the student's scores.
- We'll get much more into forms and form handling in PHP later, but this simple example will illustrate several of the common threads that appear in form handling in PHP (and server side scripting in general).


C:\xampp\htdocs\CNT 4714\decisions.html - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
decisions.html
Hyper Text Markup Langua
length : 819 lines : 18
Ln : 1 Col : 1 Sel : 0
Dos\Windows
ANSI
INS

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Grade Calculations</title>
    <meta http-equi="content-type" content="text/html' charset=iso-8859-1" />
</head>
<body  style = "font-family: arial, sans-serif;
       background-color: #856363"  background=image1.jpg>
    <form action="decisionsWithGlobals.php" method="post" >
        <font size=4 color=blue> Please Enter Scores </font> <br />
        Enter First Score <input type="text" size="4" maxlength="7" name="grade1" > <br />
        Enter Second Score <input type="text" size="4" maxlength="7" name="grade2" > <br />
        <input type="submit" value="Click To Submit" >
        <input type="reset" value="Clear And Restart" >
    </form>
</body>
</html>
```

decisions.html

Controlling Script Flow In PHP



Please Enter Scores

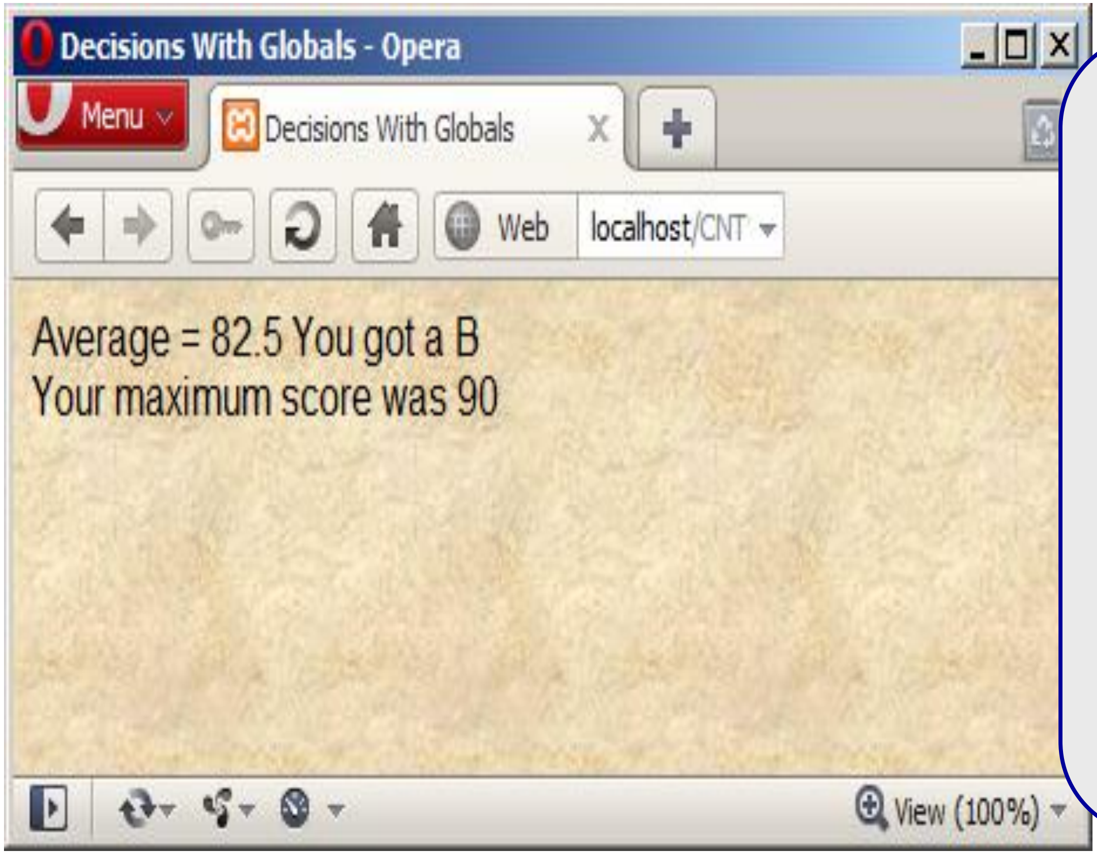
Enter First Score 75

Enter Second Score 90

Executing `decisions.html`

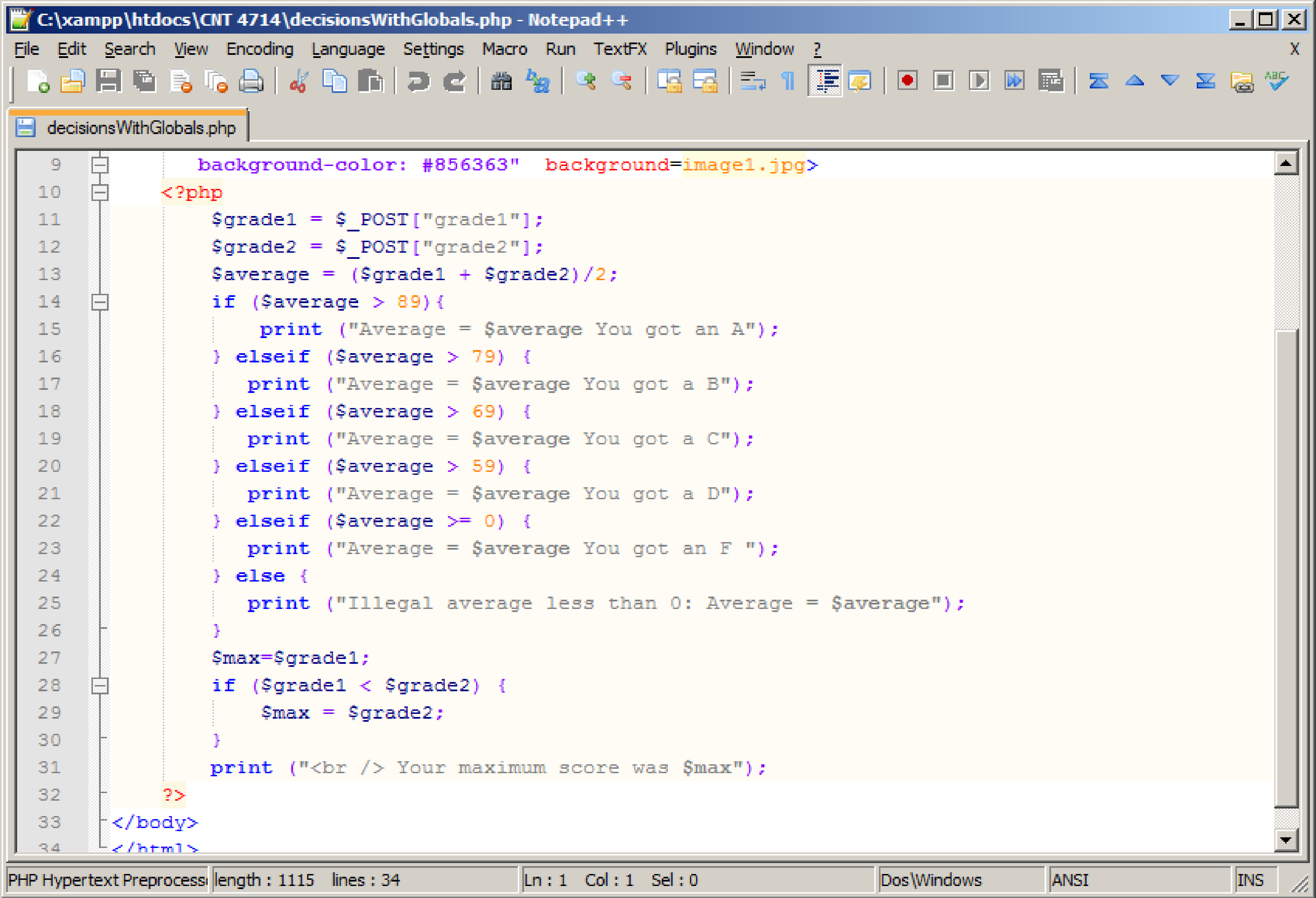
User enters two scores, clicks submit button.

Controlling Script Flow In PHP



```
Average = 82.5 You got a B
Your maximum score was 90
```

Clicking the submit button triggers the action of the form and invokes the script `decisions.php` The script generates this page. The PHP script is shown on the next page.



```
       background-color: #856363"  background=image1.jpg>
    <?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 ("Average = $average You got an F ");
        } else {
           print ("Illegal average less than 0: Average = $average");
        }
        $max=$grade1;
        if ($grade1 < $grade2) {
            $max = $grade2;
        }
        print ("<br /> Your maximum score was $max");
    ?>
</body>
</html>
```

Controlling Script Flow In PHP

- PHP supports three types of iterative constructs:
  - the `while` loop (both top and bottom tested versions are supported)
  - the `for` loop
  - and the `foreach` loop.
- The `for` and `while` loops act as you would expect given your knowledge of other programming languages. The `foreach` loop applies specifically to arrays in PHP. We’ll look at the `foreach` loop later.
- The next couple of pages show the basic syntax for each of the iterative constructs in PHP.

Controlling Script Flow In PHP

- The syntax for the top tested version of the `while` loop is:

```
while (expression) {
        //statements to execute
}
```

- The syntax for the bottom tested version of the `while` loop is:

```
do {
        //statements to execute
} while (expression);
```

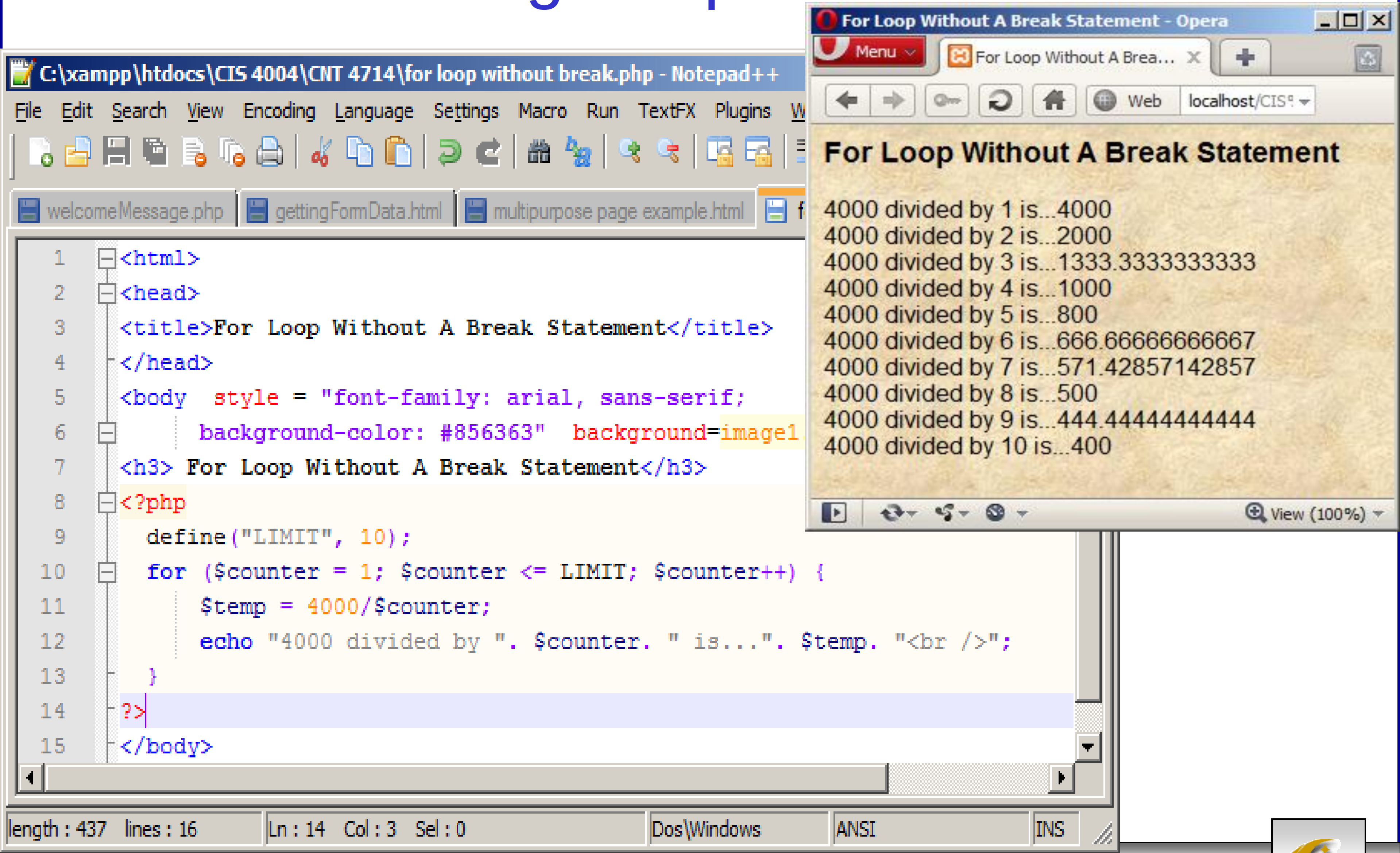
Controlling Script Flow In PHP

- The basic syntax `for` the for statement is:

```
for (initialization expr; test expr; modifying expr) {
        //statements to be executed
}
```

- The next couple of pages illustrates some of the nuances of dealing with counted loops in PHP.

Controlling Script Flow In PHP

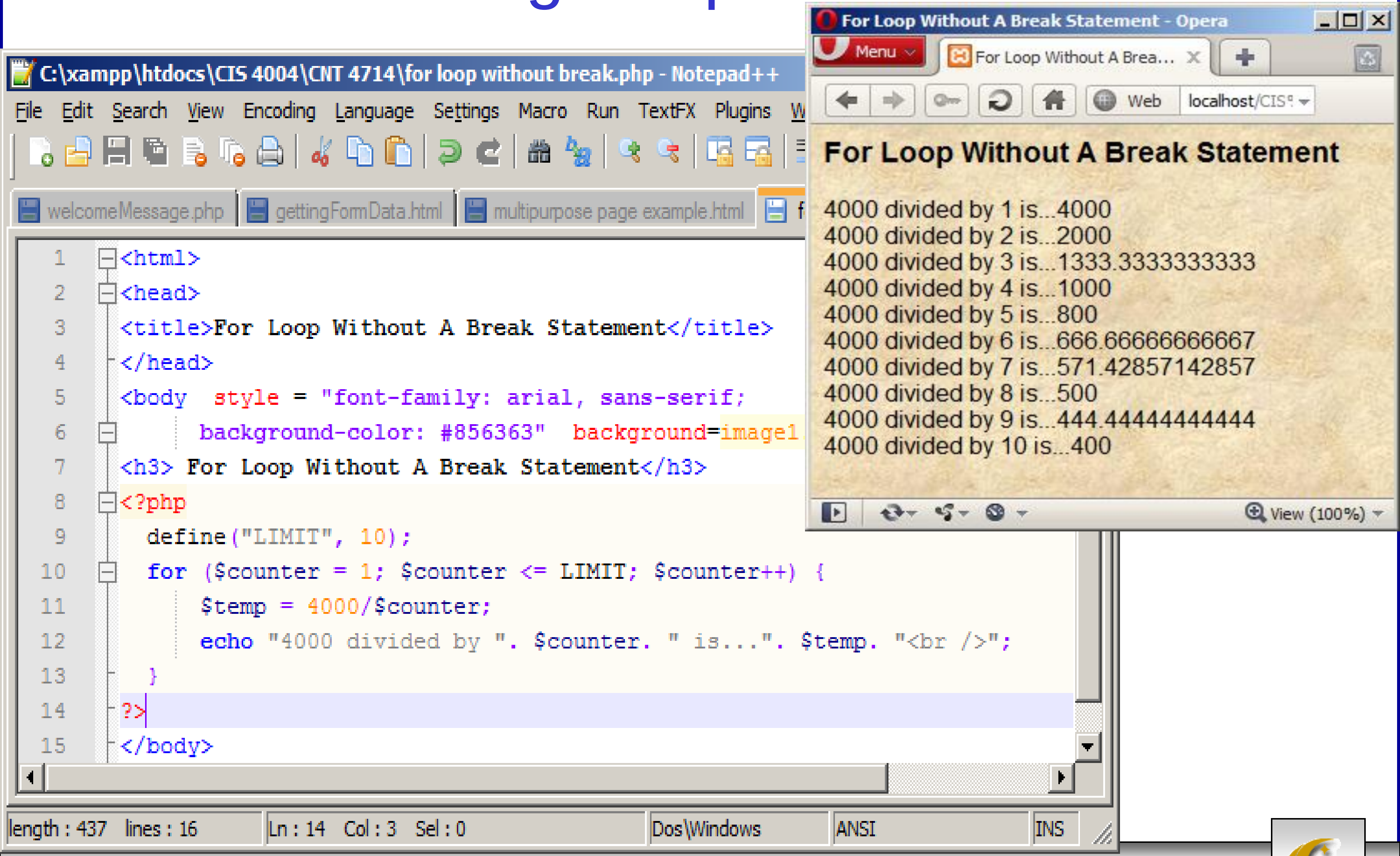

C:\xampp\htdocs\CIS 4004\CNT 4714\for loop without break.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins W
welcomeMessage.php
gettingFormData.html
multipurpose page example.html
length : 437 lines : 16
Ln : 14 Col : 3 Sel : 0
Dos\Windows
ANSI
INS
For Loop Without A Break Statement - Opera
Menu
For Loop Without A Brea...
Web
localhost/CIS
View (100%)

```
<html>
<head>
<title>For Loop Without A Break Statement</title>
</head>
<body  style = "font-family: arial, sans-serif;
       background-color: #856363"  background=image1
<h3> For Loop Without A Break Statement</h3>
<?php
  define("LIMIT", 10);
  for ($counter = 1; $counter <= LIMIT; $counter++) {
      $temp = 4000/$counter;
      echo "4000 divided by ". $counter. " is...". $temp. "<br />";
  }
?>
</body>
```

For Loop Without A Break Statement

```
4000 divided by 1 is...4000
4000 divided by 2 is...2000
4000 divided by 3 is...1333.3333333333
4000 divided by 4 is...1000
4000 divided by 5 is...800
4000 divided by 6 is...666.66666666667
4000 divided by 7 is...571.42857142857
4000 divided by 8 is...500
4000 divided by 9 is...444.44444444444
4000 divided by 10 is...400
```

Controlling Script Flow In PHP

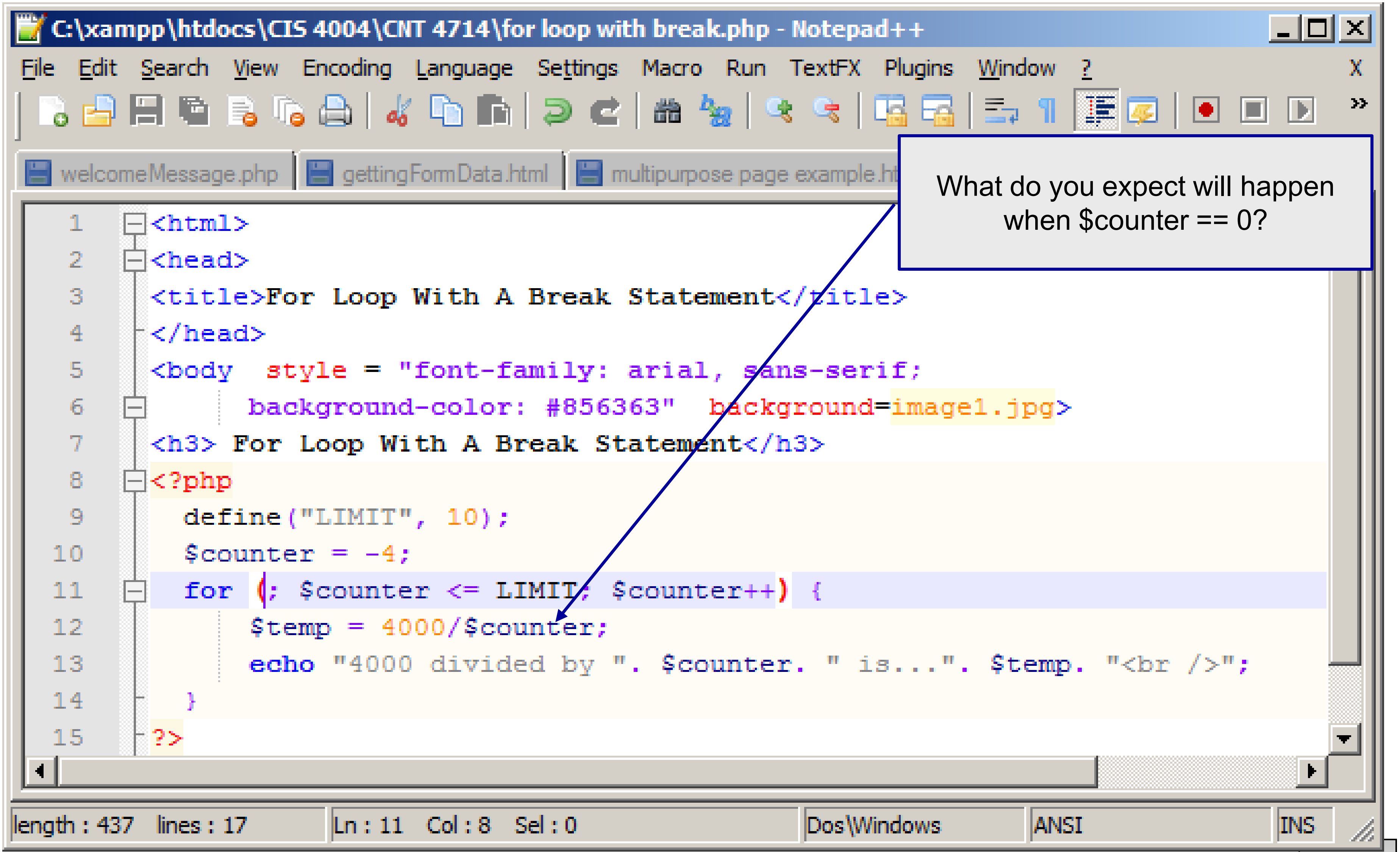

C:\xampp\htdocs\CIS 4004\CNT 4714\for loop without break.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins W
welcomeMessage.php
gettingFormData.html
multipurpose page example.html
length : 437 lines : 16
Ln : 14 Col : 3 Sel : 0
Dos\Windows
ANSI
INS
For Loop Without A Break Statement - Opera
Menu
For Loop Without A Brea...
Web
localhost/CIS
View (100%)

```
<html>
<head>
<title>For Loop Without A Break Statement</title>
</head>
<body  style = "font-family: arial, sans-serif;
      background-color: #856363"  background=image1
<h3> For Loop Without A Break Statement</h3>
<?php
  define("LIMIT", 10);
  for ($counter = 1; $counter <= LIMIT; $counter++) {
      $temp = 4000/$counter;
      echo "4000 divided by ". $counter. " is...". $temp. "<br />";
  }
?>
</body>
```

For Loop Without A Break Statement

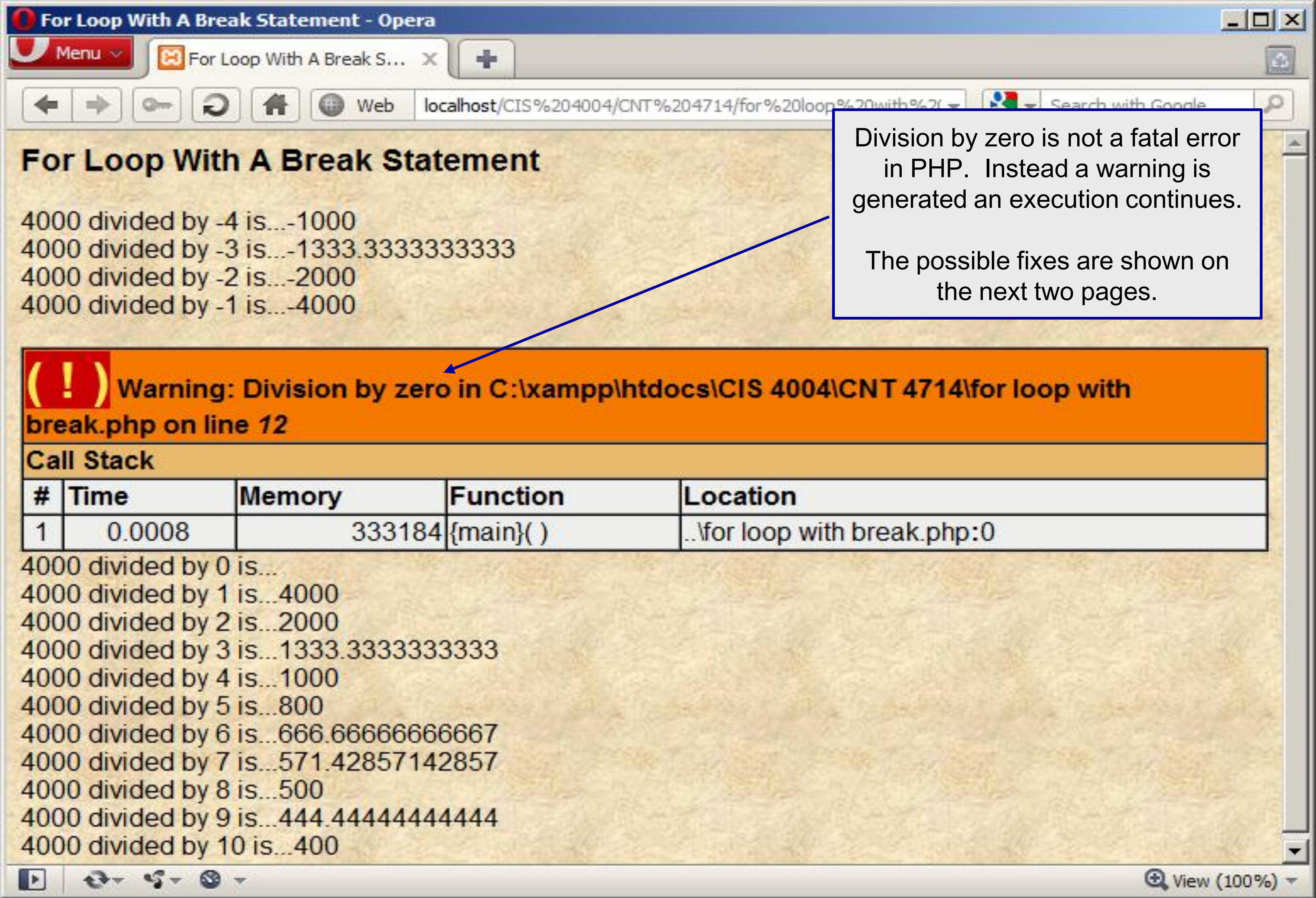
```
4000 divided by 1 is...4000
4000 divided by 2 is...2000
4000 divided by 3 is...1333.3333333333
4000 divided by 4 is...1000
4000 divided by 5 is...800
4000 divided by 6 is...666.66666666667
4000 divided by 7 is...571.42857142857
4000 divided by 8 is...500
4000 divided by 9 is...444.44444444444
4000 divided by 10 is...400
```

Controlling Script Flow In PHP



```
<html>
<head>
<title>For Loop With A Break Statement</title>
</head>
<body  style = "font-family: arial, sans-serif;
      background-color: #856363"  background=image1.jpg>
<h3> For Loop With A Break Statement</h3>
<?php
  define("LIMIT", 10);
  $counter = -4;
  for (; $counter <= LIMIT; $counter++) {
      $temp = 4000/$counter;
      echo "4000 divided by ". $counter. " is...". $temp. "<br />";
  }
?>
```

What do you expect will happen when $counter == 0?


For Loop With A Break Statement - Opera
Menu
For Loop With A Break S...
Web
localhost/CIS%204004/CNT%204714/for%20loop%20with%2(
Search with Google
View (100%)

For Loop With A Break Statement

```
4000 divided by -4 is...-1000
4000 divided by -3 is...-1333.3333333333
4000 divided by -2 is...-2000
4000 divided by -1 is...-4000
```

(!) Warning: Division by zero in C:\xampp\htdocs\CIS 4004\CNT 4714\for loop with break.php on line *12*

Call Stack

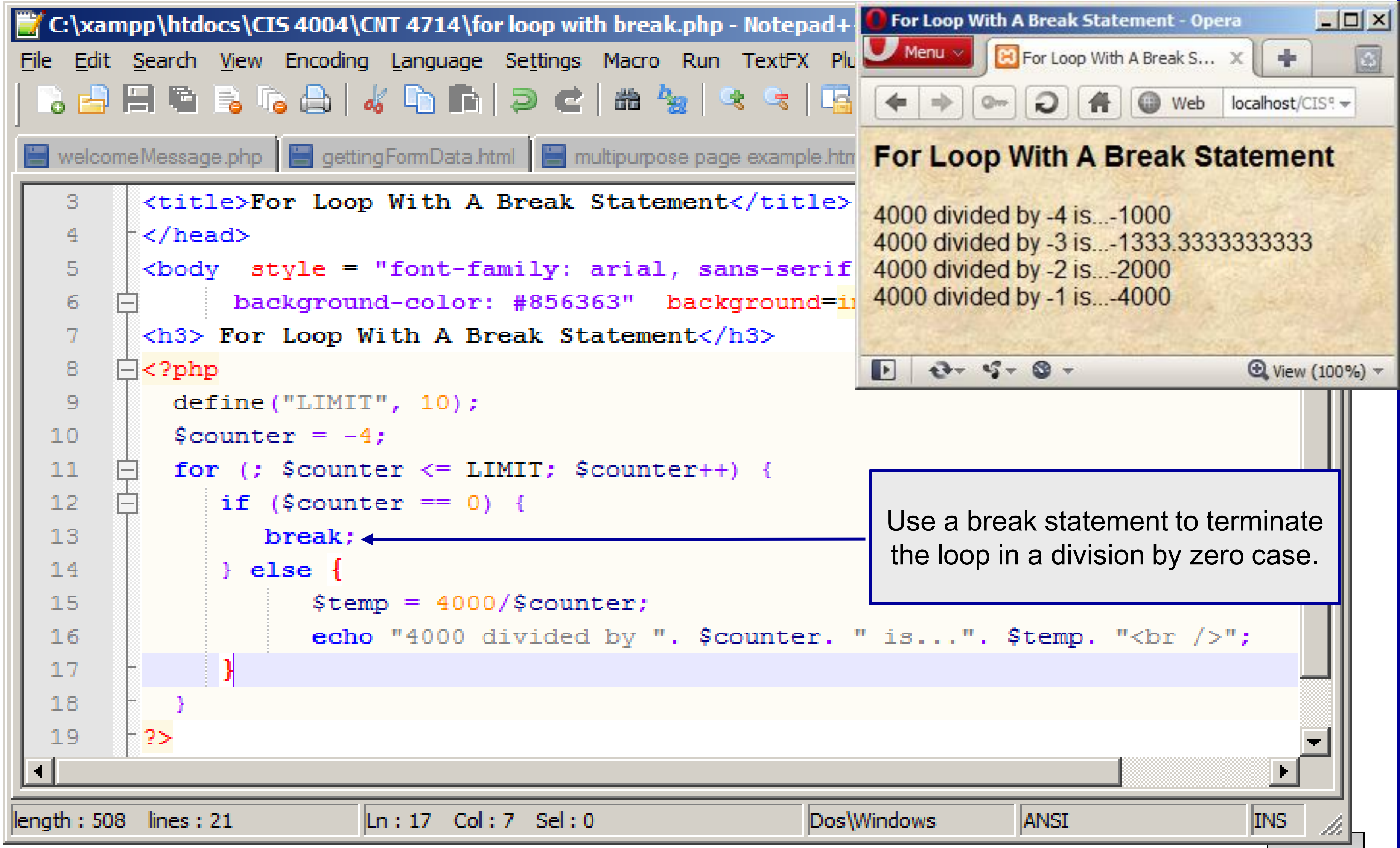
#	Time	Memory	Function	Location
1	0.0008	333184	{main}( )	..\for loop with break.php:0

```
4000 divided by 0 is...
4000 divided by 1 is...4000
4000 divided by 2 is...2000
4000 divided by 3 is...1333.3333333333
4000 divided by 4 is...1000
4000 divided by 5 is...800
4000 divided by 6 is...666.66666666667
4000 divided by 7 is...571.42857142857
4000 divided by 8 is...500
4000 divided by 9 is...444.44444444444
4000 divided by 10 is...400
```

Division by zero is not a fatal error in PHP. Instead a warning is generated an execution continues.

The possible fixes are shown on the next two pages.

Controlling Script Flow In PHP


C:\xampp\htdocs\CIS 4004\CNT 4714\for loop with break.php - Notepad+
File
Edit
Search
View
Encoding
Language
Settings
Macro
Run
TextFX
Plu
welcomeMessage.php
gettingFormData.html
multipurpose page example.htm
For Loop With A Break Statement - Opera
Menu
For Loop With A Break S...
Web
localhost/CIS
View (100%)

```
<title>For Loop With A Break Statement</title>
</head>
<body  style = "font-family: arial, sans-serif
      background-color: #856363"  background=i
<h3> For Loop With A Break Statement</h3>
<?php
  define("LIMIT", 10);
  $counter = -4;
  for (; $counter <= LIMIT; $counter++) {
     if ($counter == 0) {
        break;
     } else {
        $temp = 4000/$counter;
        echo "4000 divided by ". $counter. " is...". $temp. "<br />";
     }
  }
?>
```

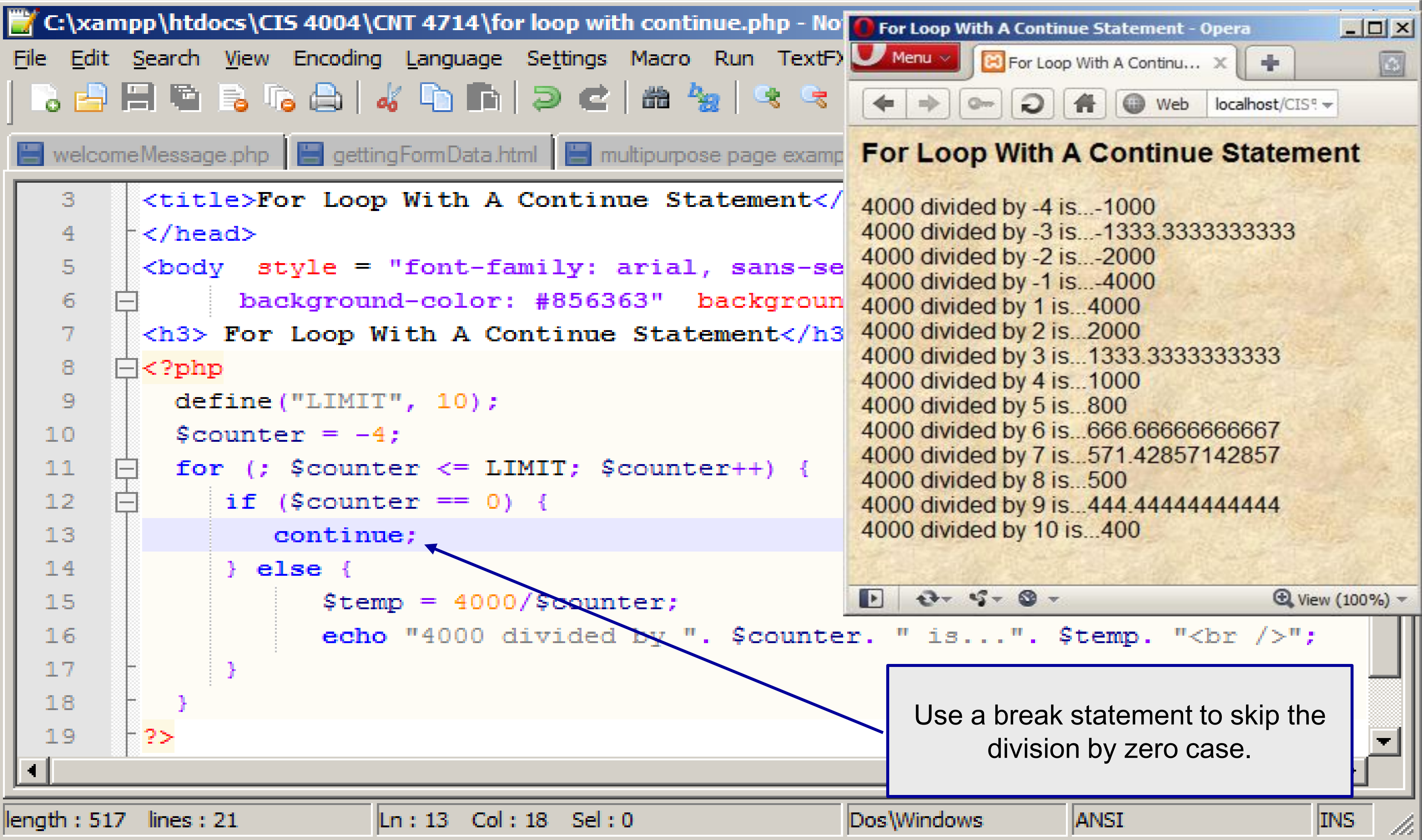
length : 508   lines : 21   Ln : 17   Col : 7   Sel : 0   Dos\Windows   ANSI   INS

For Loop With A Break Statement

```
4000 divided by -4 is...-1000
4000 divided by -3 is...-1333.3333333333
4000 divided by -2 is...-2000
4000 divided by -1 is...-4000
```

Use a break statement to terminate the loop in a division by zero case.

Controlling Script Flow In PHP

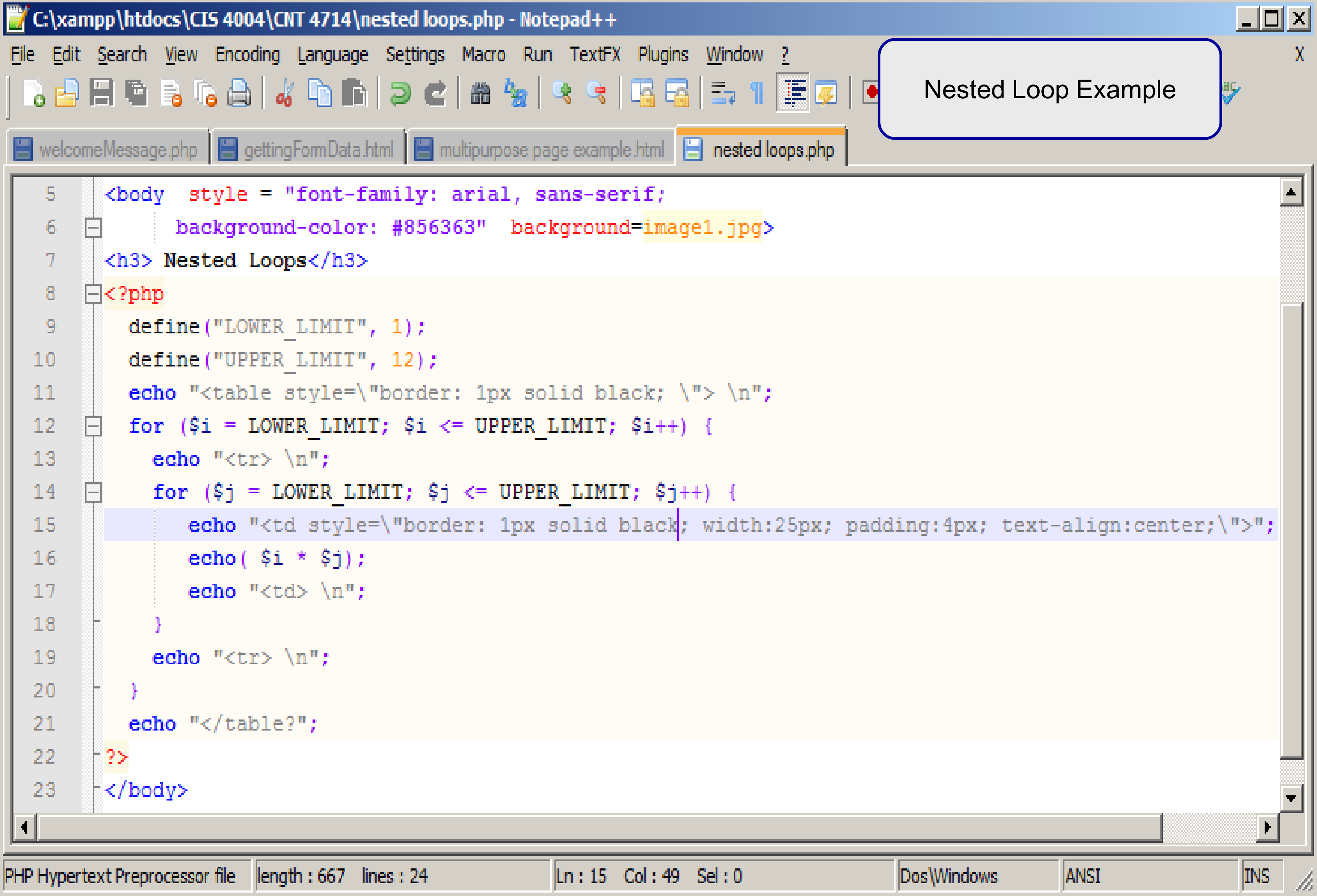

C:\xampp\htdocs\CIS 4004\CNT 4714\for loop with continue.php - No
File Edit Search View Encoding Language Settings Macro Run TextF
welcomeMessage.php
gettingFormData.html
multipurpose page examp
length : 517 lines : 21
Ln : 13 Col : 18 Sel : 0
Dos\Windows
ANSI
INS
For Loop With A Continue Statement - Opera
Menu
For Loop With A Continu...
Web
localhost/CIS
View (100%)

```
<title>For Loop With A Continue Statement</
</head>
<body  style = "font-family: arial, sans-se
      background-color: #856363"  backgroun
<h3> For Loop With A Continue Statement</h3
<?php
  define("LIMIT", 10);
  $counter = -4;
  for (; $counter <= LIMIT; $counter++) {
     if ($counter == 0) {
        continue;
     } else {
        $temp = 4000/$counter;
        echo "4000 divided by ". $counter. " is...". $temp. "<br />";
     }
  }
?>
```

For Loop With A Continue Statement

```
4000 divided by -4 is...-1000
4000 divided by -3 is...-1333.3333333333
4000 divided by -2 is...-2000
4000 divided by -1 is...-4000
4000 divided by 1 is...4000
4000 divided by 2 is...2000
4000 divided by 3 is...1333.3333333333
4000 divided by 4 is...1000
4000 divided by 5 is...800
4000 divided by 6 is...666.66666666667
4000 divided by 7 is...571.42857142857
4000 divided by 8 is...500
4000 divided by 9 is...444.44444444444
4000 divided by 10 is...400
```

Use a break statement to skip the division by zero case.

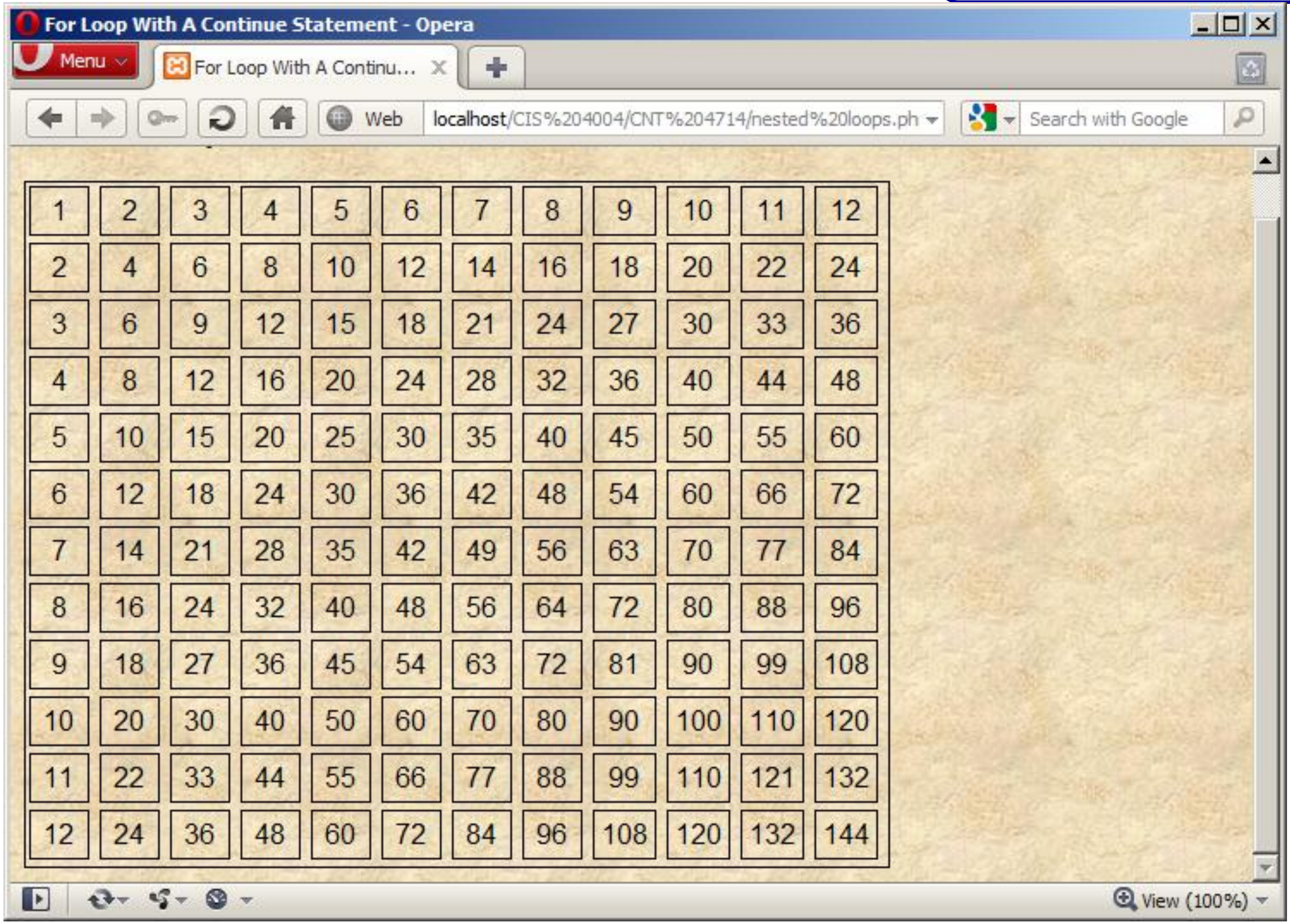

C:\xampp\htdocs\CIS 4004\CNT 4714\nested loops.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
welcomeMessage.php
gettingFormData.html
multipurpose page example.html
nested loops.php

Nested Loop Example

```
<body  style = "font-family: arial, sans-serif;
       background-color: #856363"  background=image1.jpg>
<h3> Nested Loops</h3>
<?php
  define("LOWER_LIMIT", 1);
  define("UPPER_LIMIT", 12);
  echo "<table style=\"border: 1px solid black; \"> \n";
  for ($i = LOWER_LIMIT; $i <= UPPER_LIMIT; $i++) {
    echo "<tr> \n";
    for ($j = LOWER_LIMIT; $j <= UPPER_LIMIT; $j++) {
       echo "<td style=\"border: 1px solid black; width:25px; padding:4px; text-align:center;\">";
       echo( $i * $j);
       echo "<td> \n";
    }
    echo "<tr> \n";
  }
  echo "</table?";
?>
</body>
```

PHP Hypertext Preprocessor file | length : 667 lines : 24 | Ln : 15 Col : 49 Sel : 0 | Dos\Windows | ANSI | INS

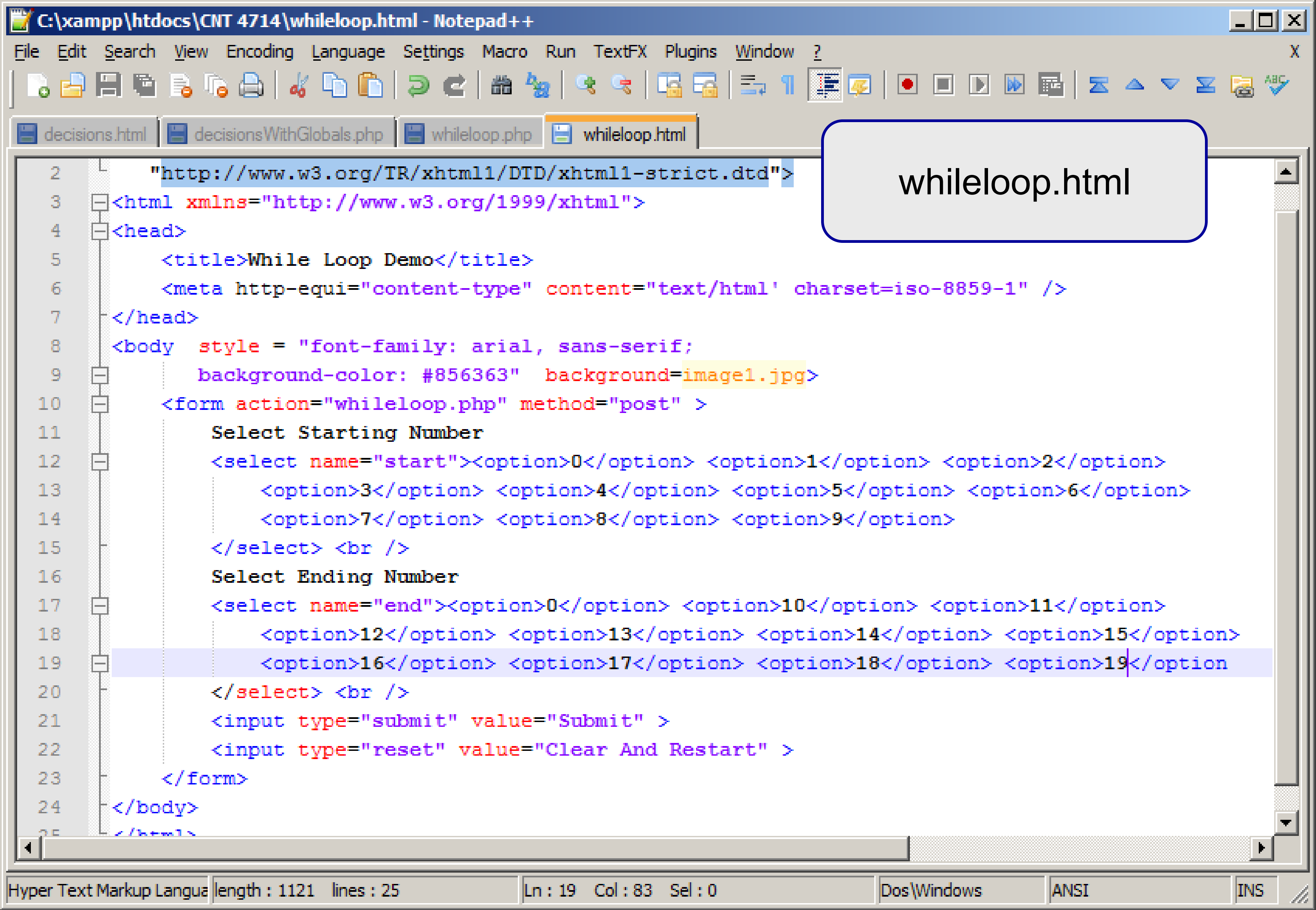
Nested Loop Example - Output


For Loop With A Continue Statement - Opera
Menu
For Loop With A Continu...
Web
localhost/CIS%204004/CNT%204714/nested%20loops.ph
Search with Google
View (100%)

1	2	3	4	5	6	7	8	9	10	11	12
2	4	6	8	10	12	14	16	18	20	22	24
3	6	9	12	15	18	21	24	27	30	33	36
4	8	12	16	20	24	28	32	36	40	44	48
5	10	15	20	25	30	35	40	45	50	55	60
6	12	18	24	30	36	42	48	54	60	66	72
7	14	21	28	35	42	49	56	63	70	77	84
8	16	24	32	40	48	56	64	72	80	88	96
9	18	27	36	45	54	63	72	81	90	99	108
10	20	30	40	50	60	70	80	90	100	110	120
11	22	33	44	55	66	77	88	99	110	121	132
12	24	36	48	60	72	84	96	108	120	132	144

Controlling Script Flow In PHP

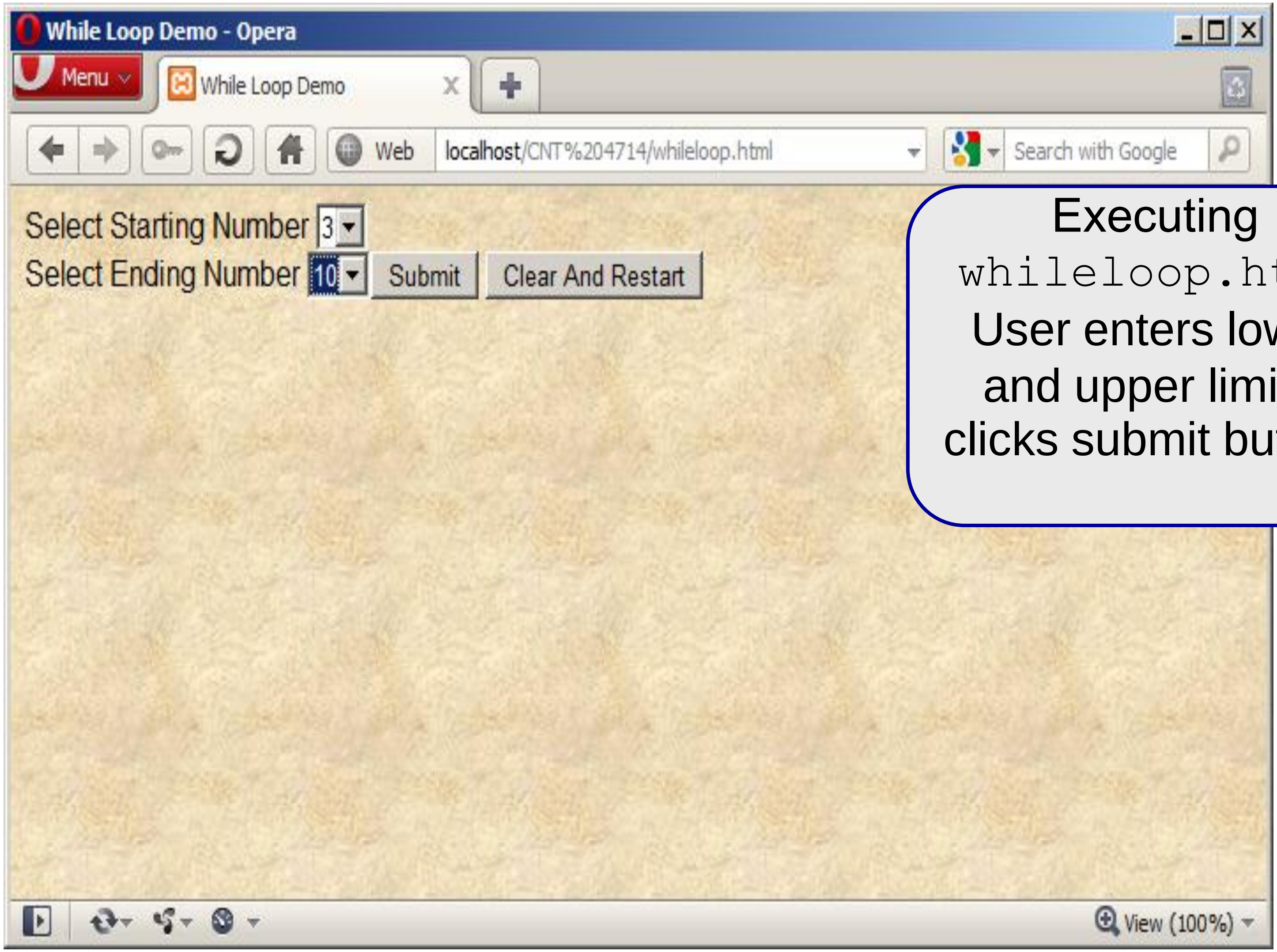
- The example on the next couple of pages illustrates a while loop. Again, I’ve used a form to extract user input. This time the user input sets the lower and upper limit on the loop.



whileloop.html

```
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>While Loop Demo</title>
    <meta http-equi="content-type" content="text/html' charset=iso-8859-1" />
</head>
<body  style = "font-family: arial, sans-serif;
        background-color: #856363"  background=image1.jpg>
    <form action="whileloop.php" method="post" >
        Select Starting Number
        <select name="start"><option>0</option> <option>1</option> <option>2</option>
            <option>3</option> <option>4</option> <option>5</option> <option>6</option>
            <option>7</option> <option>8</option> <option>9</option>
        </select> <br />
        Select Ending Number
        <select name="end"><option>0</option> <option>10</option> <option>11</option>
            <option>12</option> <option>13</option> <option>14</option> <option>15</option>
            <option>16</option> <option>17</option> <option>18</option> <option>19</option
        </select> <br />
        <input type="submit" value="Submit" >
        <input type="reset" value="Clear And Restart" >
    </form>
</body>
```

Controlling Script Flow In PHP



Select Starting Number 3

Select Ending Number 10

Submit

Clear And Restart

Executing `whileloop.html` User enters lower and upper limits, clicks submit button.

Controlling Script Flow In PHP

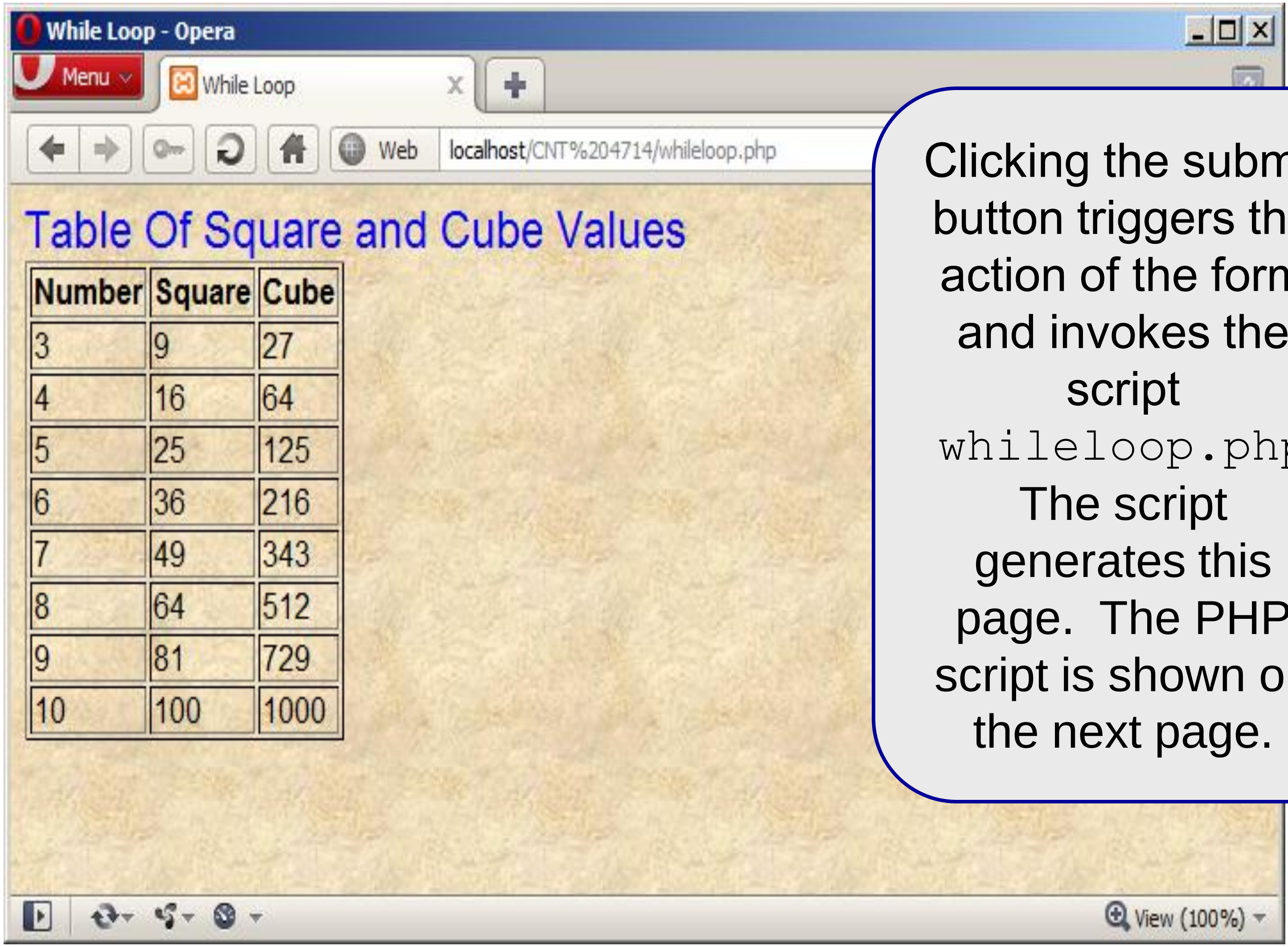
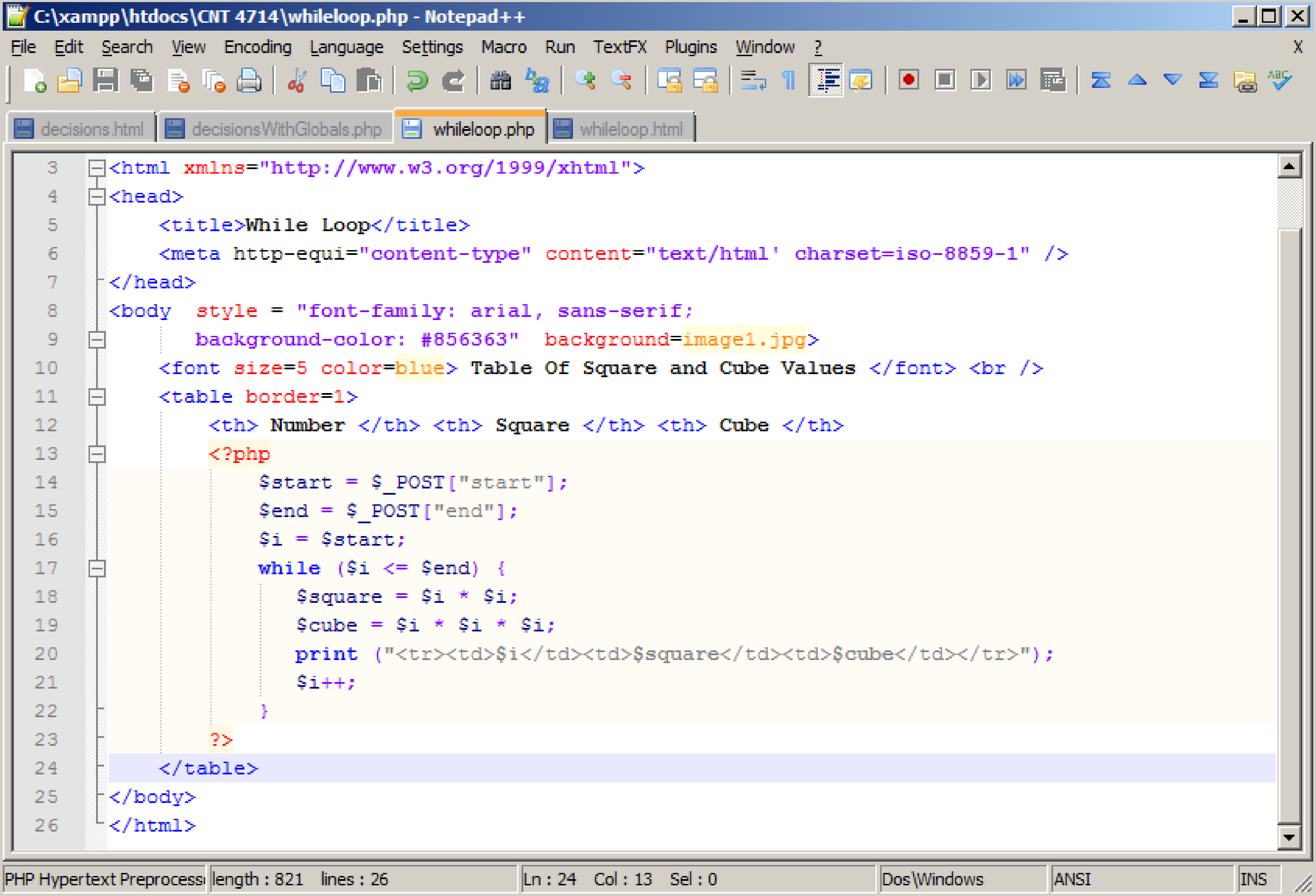


Table Of Square and Cube Values

Number	Square	Cube
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

Clicking the submit button triggers the action of the form and invokes the script `whileloop.php` The script generates this page. The PHP script is shown on the next page.

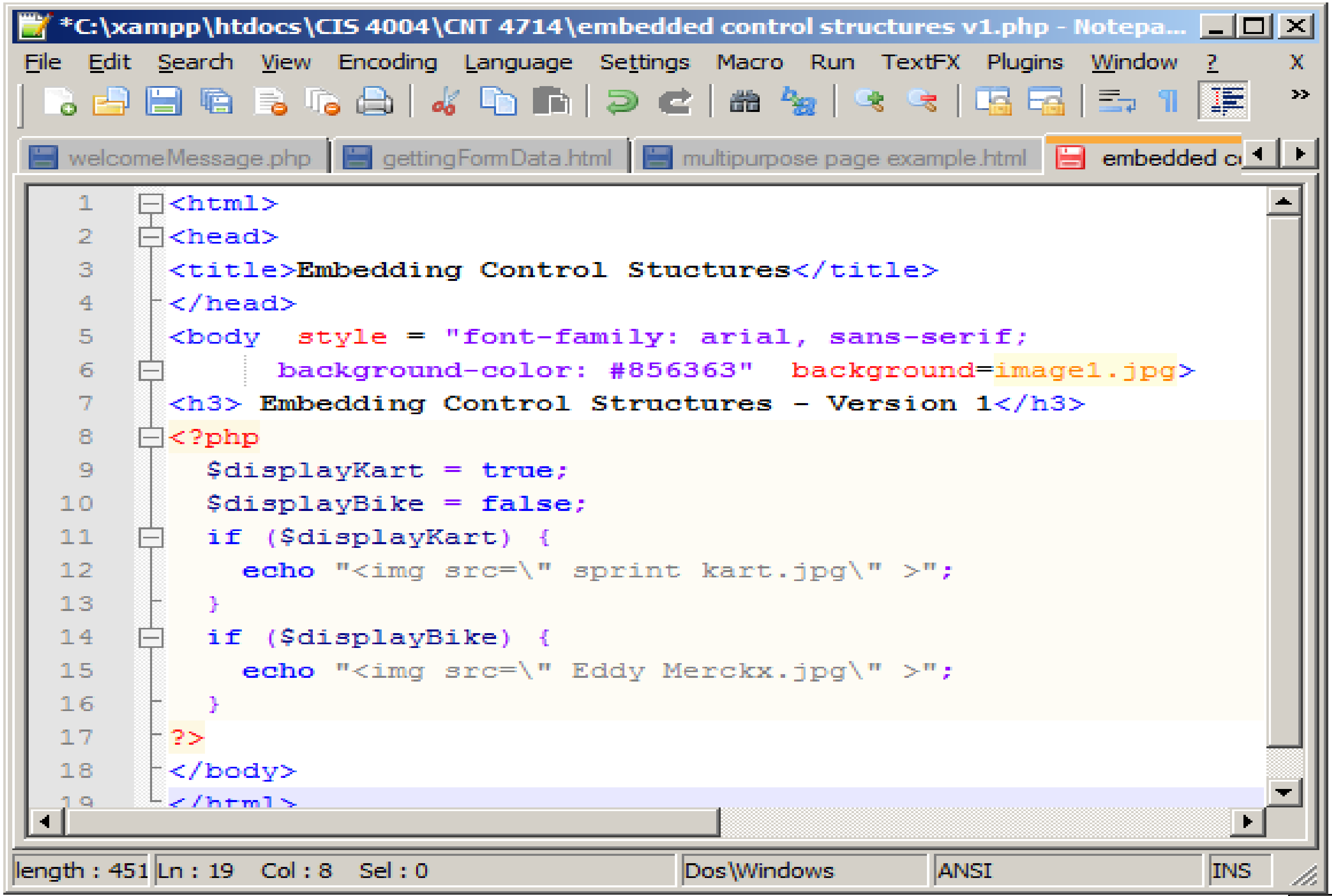


```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>While Loop</title>
    <meta http-equi="content-type" content="text/html' charset=iso-8859-1" />
</head>
<body  style = "font-family: arial, sans-serif;
       background-color: #856363"  background=image1.jpg>
    <font size=5 color=blue> Table Of Square and Cube Values </font> <br />
    <table border=1>
        <th> Number </th> <th> Square </th> <th> Cube </th>
        <?php
            $start = $_POST["start"];
            $end = $_POST["end"];
            $i = $start;
            while ($i <= $end) {
               $square = $i * $i;
               $cube = $i * $i * $i;
               print ("<tr><td>$i</td><td>$square</td><td>$cube</td></tr>");
               $i++;
            }
        ?>
    </table>
</body>
</html>
```

Embedding Control Structures

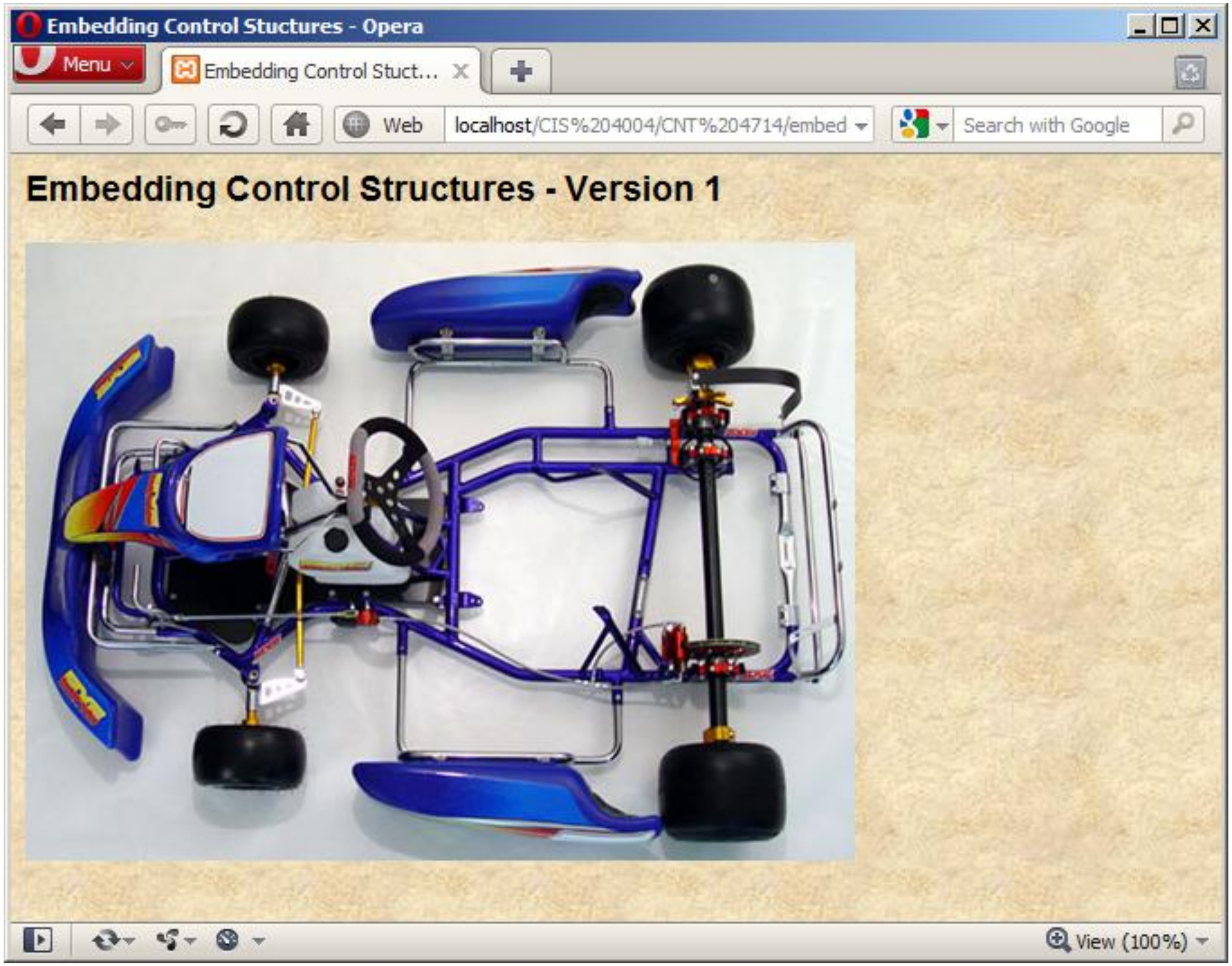
- Now that we’ve seen most of the control structures in PHP, we need to see how these control structures can be used more effectively to produce XHTML elements (or any other output).
- PHP is an embedded language that enables you to code both your XHTML and the supporting script in the same document.
- PHP takes this concept a bit further by allowing you to “turn off” the PHP parser during a control structure and embed non-PHP output without losing the logic provided by the control structure.
- The following example, illustrates this concept by displaying an image in your XHTML document only when a variable is set to true.

Embedding Control Structures


*C:\xampp\htdocs\CIS 4004\CNT 4714\embedded control structures v1.php - Notepa...
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
welcomeMessage.php
gettingFormData.html
multipurpose page example.html
embedded c
length : 451 Ln : 19 Col : 8 Sel : 0
Dos\Windows
ANSI
INS

```
<html>
<head>
<title>Embedding Control Stuctures</title>
</head>
<body  style = "font-family: arial, sans-serif;
       background-color: #856363"  background=image1.jpg>
<h3> Embedding Control Structures - Version 1</h3>
<?php
  $displayKart = true;
  $displayBike = false;
  if ($displayKart) {
    echo "<img src=\" sprint kart.jpg\" >";
  }
  if ($displayBike) {
    echo "<img src=\" Eddy Merckx.jpg\" >";
  }
?>
</body>
</html>
```

Embedding Control Structures



Embedding Control Stuctures - Opera
Menu
Embedding Control Stuct...
Web
localhost/CIS%204004/CNT%204714/embed
Search with Google
View (100%)

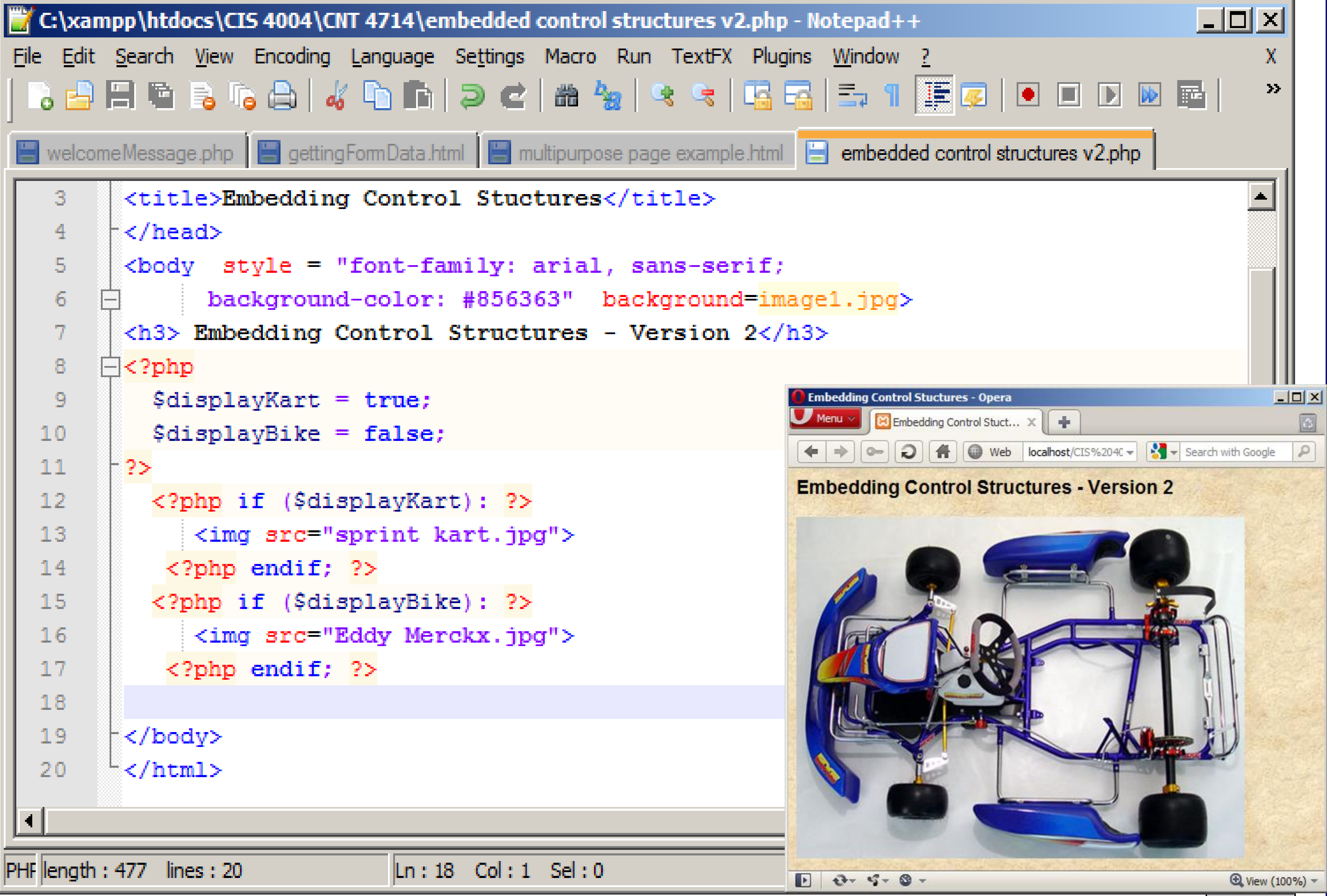
Embedding Control Structures - Version 1

Embedding Control Structures

- Although the previous solution works and for novice PHP programmers it seems to be the most obvious technique, PHP provides an alternate syntax that is actually allows the embedding of the control structure into the markup.
- This alternative syntax is:

```
<?php . . .
    if (conditional): ?>
        - text/whatever that should be output but not parsed
<?php endif;
 ?>
```

- This is shown in the next version of this example on the following page.


C:\xampp\htdocs\CIS 4004\CNT 4714\embedded control structures v2.php - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
welcomeMessage.php
gettingFormData.html
multipurpose page example.html
embedded control structures v2.php
Embedding Control Stuctures - Opera
Menu
Embedding Control Struct...
Web
localhost/CIS%2040
Search with Google
View (100%)

```
<title>Embedding Control Stuctures</title>
</head>
<body  style = "font-family: arial, sans-serif;
        background-color: #856363"  background=image1.jpg>
<h3> Embedding Control Structures - Version 2</h3>
<?php
  $displayKart = true;
  $displayBike = false;
?>
  <?php if ($displayKart): ?>
     <img src="sprint kart.jpg">
   <?php endif; ?>
  <?php if ($displayBike): ?>
     <img src="Eddy Merckx.jpg">
   <?php endif; ?>

</body>
</html>
```

PHP length : 477 lines : 20 Ln : 18 Col : 1 Sel : 0

Embedding Control Structures - Version 2