Lecture #10

NEWM N510: Web-Database Concepts

PHP (2)

kharrazi@iupui.edu
http://www.info510.com

Review Last Lecture

- PHP Intro
- PHP Syntax
- PHP echo
- PHP Commenting
- PHP Variables

PHP in a Nutshell

1. PHP Intro
2. PHP Syntax
3. PHP *echo*
4. PHP Comment
5. PHP Variables
6. PHP String/Array Manipulation
7. PHP Conditions
8. PHP Loops
9. PHP Functions
10. PHP Cookies/Sessions
11. PHP SSI
12. PHP Forms
13. PHP/MySQL Integration

Lecture in a Nutshell

1. PHP: String Manipulations
   - Formatting
   - Join & Split
   - Comparison
   - Match & Replace

2. PHP: Array Manipulations
   - Sorting
   - Reordering
   - Count
   - Miscellaneous

1. PHP String Manipulation

- These functions all manipulate strings in various ways. Some more specialized sections can be done by regular expressions (like Perl) and URL handling commands.

- No external libraries are needed to build this extension.

- There is no installation needed to use these functions; they are part of the PHP core.

- For regular expressions please refer to:
  http://ca.php.net/manual/en/ref.regex.php
  http://ca.php.net/manual/en/ref.pcre.php

PHP String Manipulation (cont.)

- Formatting Strings
  - Trimming, Case Changing
- Joining and Splitting Strings
- Comparing Strings
- Match and Replace Substrings
- Regular expression
  - *please refer to:*
    http://ca.php.net/manual/en/ref.regex.php
    http://ca.php.net/manual/en/ref.pcre.php

PHP String:Formatting (cont.)

- Trimming: Stripping white space (or other characters) from the beginning and/or end of a string.
- Usually we use these functions to remove additional spaces or characters that users might have typed mistakenly in the input forms.
- [] syntax means that is optional.

```
trim (string, [character list]);
ltrim (string, [character list]);
rtrim (string, [character list]);
chop (string);
```

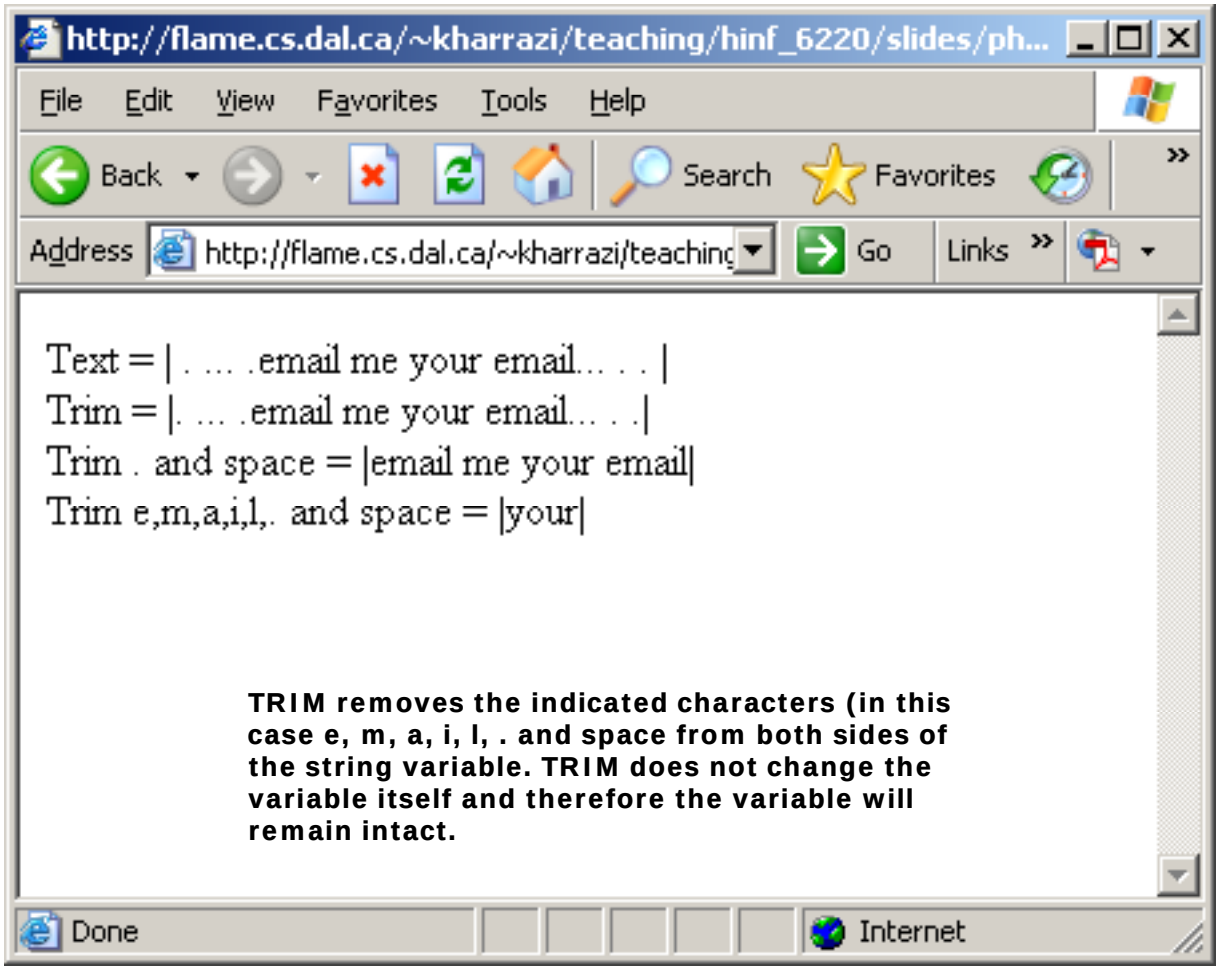
PHP String:Formatting (cont.)

trim

```
<?php
    $text = " . ... .email me your email... . . ";

    // Trim
    echo "Text = "."|".$text."|";
    echo "<br>";
    echo "Trim = "."|".trim($text)."|";
    echo "<br>";
    echo "Trim . and space = "."|".trim($text, ". ")."|";
    echo "<br>";
    echo "Trim e,m,a,i,l,. and space = "."|".trim($text,
    "email. ")."|";
?>
```

PHP String:Formatting (cont.)

trim



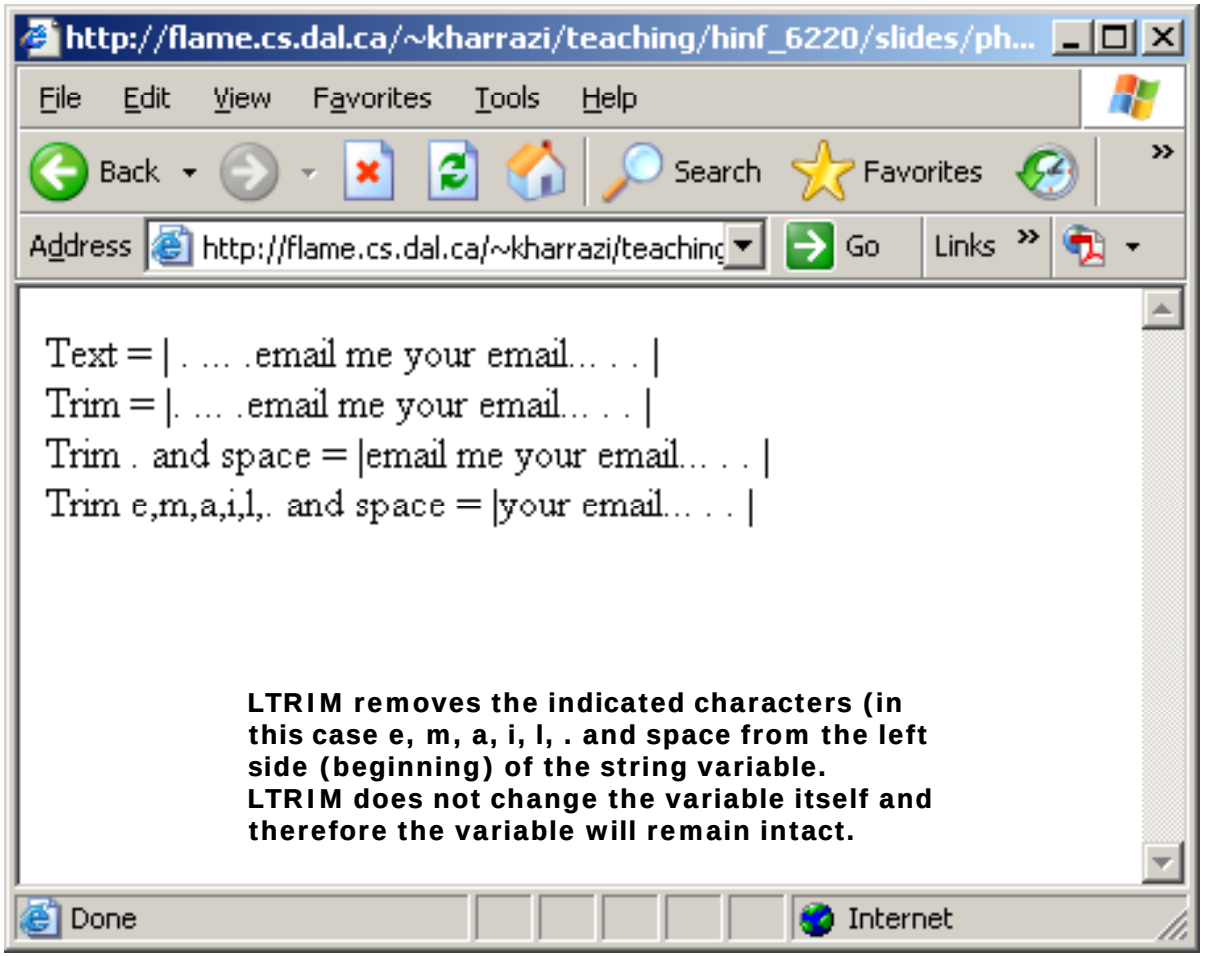
PHP String:Formatting (cont.)

ltrim

```php
<?php
    $text = " . ... .email me your email... . . ";

    // Left Trim
    echo "Text = "."|".$text."|";
    echo "<br>";
    echo "Trim = "."|".ltrim($text)."|";
    echo "<br>";
    echo "Trim . and space = "."|".ltrim($text, ". ")."|";
    echo "<br>";
    echo "Trim e,m,a,i,l,. and space = "."|".ltrim($text,
    "email. ")."|";
?>
```

PHP String:Formatting (cont.)

ltrim



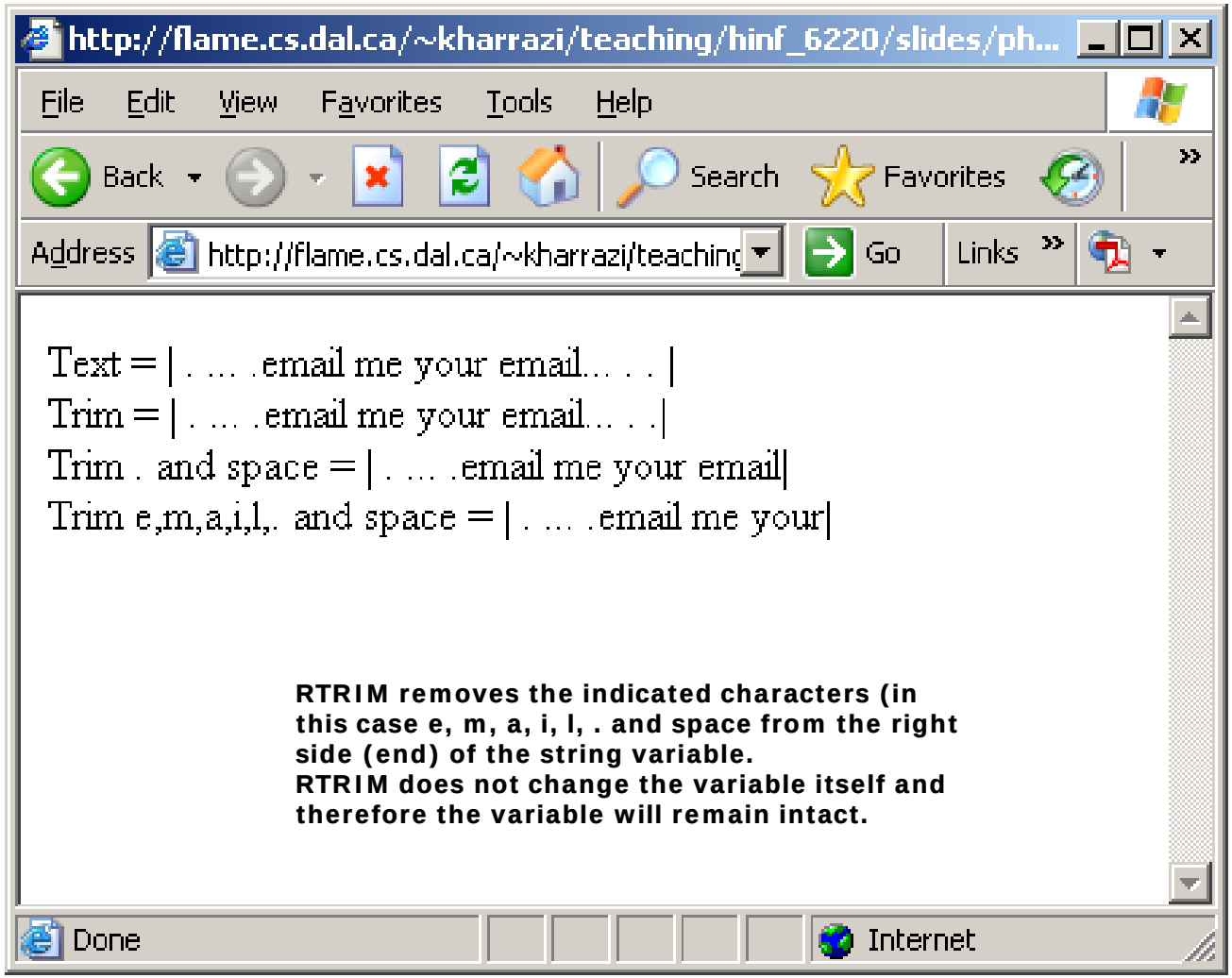
PHP String:Formatting (cont.)

rtrim

```php
<?php
    $text = " . ... .email me your email... . . ";

    // Right Trim
    echo "Text = "."|".$text."|";
    echo "<br>";
    echo "Trim = "."|".rtrim($text)."|";
    echo "<br>";
    echo "Trim . and space = "."|".rtrim($text, ". ")."|";
    echo "<br>";
    echo "Trim e,m,a,i,l,. and space = "."|".rtrim($text,
    "email. ")."|";
?>
```

PHP String:Formatting (cont.)

rtrim



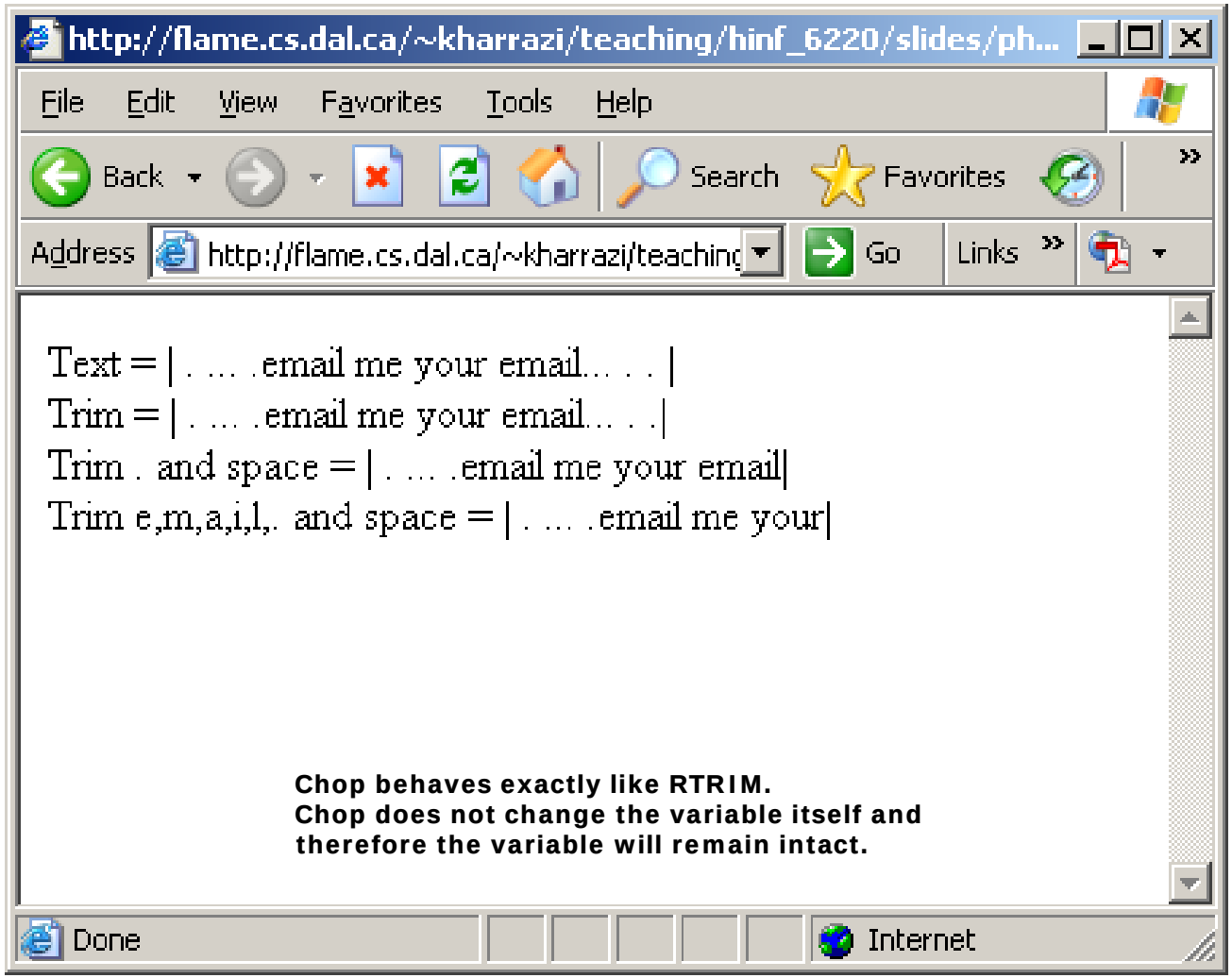
PHP String:Formatting (cont.)

chop

```
<?php
    $text = " . ... .email me your email... . . ";

    // Chop
    echo "Text = "."|".$text."|";
    echo "<br>";
    echo "Trim = "."|".chop($text)."|";
    echo "<br>";
    echo "Trim . and space = "."|".chop($text, ". ")."|";
    echo "<br>";
    echo "Trim e,m,a,i,l,. and space = "."|".chop($text,
    "email. ")."|";
?>
```

PHP String:Formatting (cont.)

chop



PHP String:Formatting (cont.)

- Case Changing: These functions will change the case of a particular character to lower or upper.
- Usually we use these functions to format the users input data received by forms.
- Remember that UNIX systems are case sensitive!

```
strtoupper (string);
strtolower (string);
ucfirst (string);
ucwords (string);
```

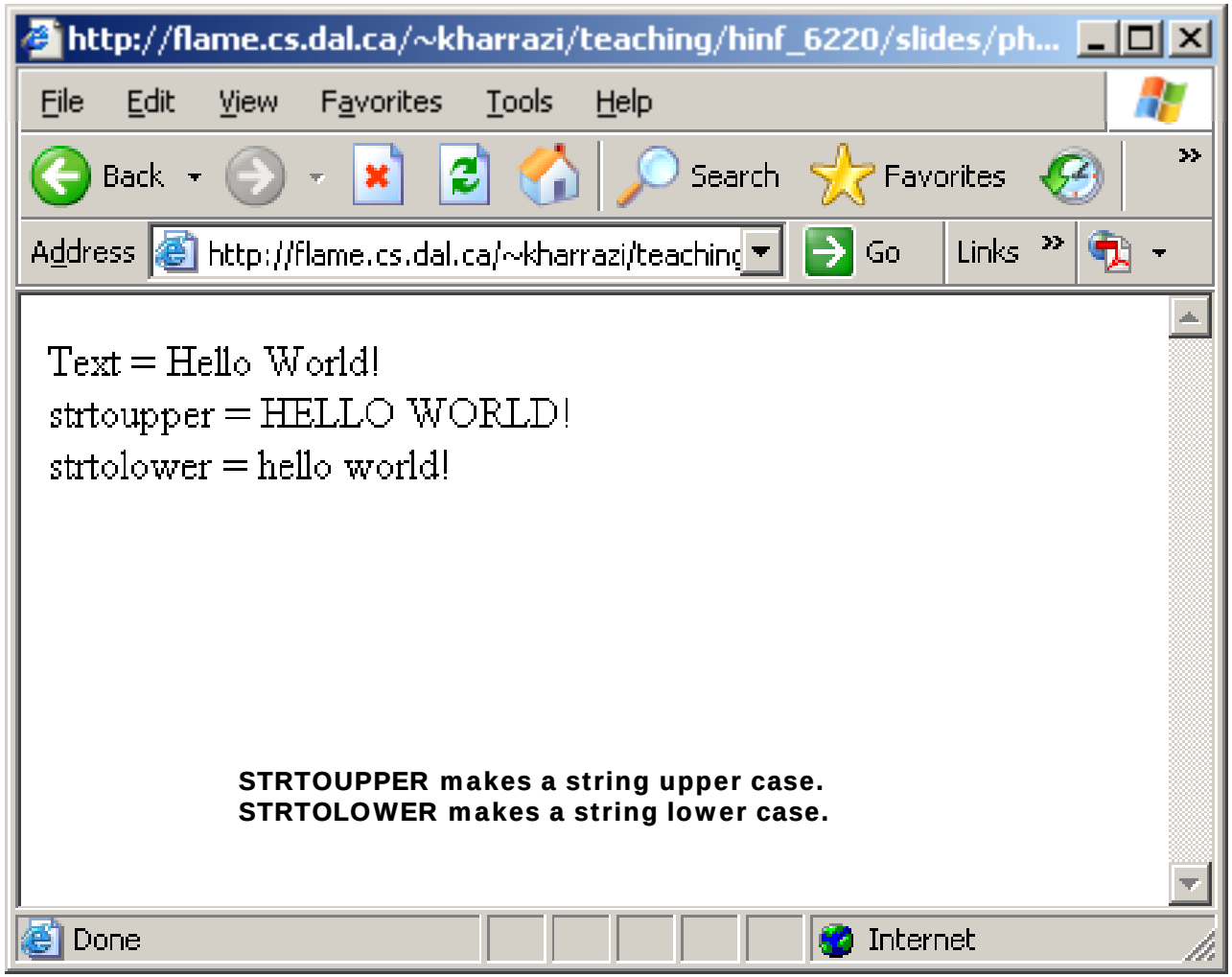
PHP String:Formatting (cont.)

strtoupper
strtolower

```php
<?php
    $text = "Hello World!";
    echo "Text = ".$text;

    // strtoupper changes all of the characters to upper case
    echo "<br>";
    echo "strtoupper = ".strtoupper($text);

    // strtoupper changes all of the characters to upper case
    echo "<br>";
    echo "strtolower = ".strtolower($text);
?>
```

PHP String:Formatting (cont.)

strtoupper
strtolower



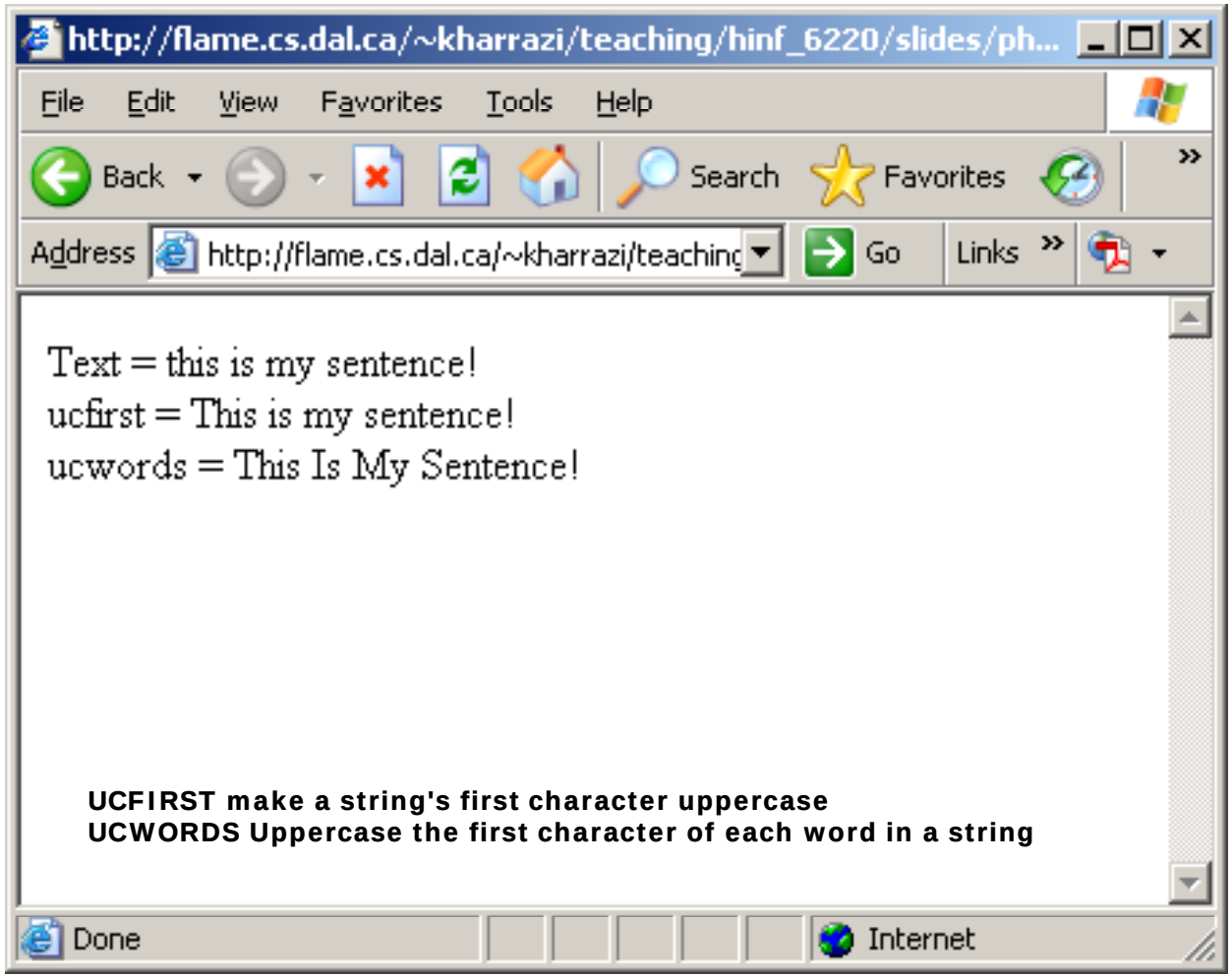
PHP String:Formatting (cont.)

ucfirst
ucwords

```php
<?php
    $text = "this is my sentence!";
    echo "Text = ".$text;

    // ucfirst make a string's first character uppercase
    echo "<br>";
    echo "ucfirst = ".ucfirst($text);

    // ucwords Uppercase the first character of each word in a
    string
    echo "<br>";
    echo "ucwords = ".ucwords($text);
?>
```

PHP String:Formatting (cont.)

ucfirst
ucwords



PHP String:Formatting (cont.)

- These commands are recommended for reading:

  print()
  printf();
  sprintf();
  print_r();
  nl2br();
  addslashes();
  stripslashes();

- Please refer to the following link for more information:
  http://ca.php.net/manual/en/ref.strings.php

PHP String:Join&Split

- Joining: These functions will join elements of an array to make a string.

```
implode (seperator, array elements);
join (seperator, array elements);
```

- Splitting: These functions will split a string and create an array from the fragments.

```
explode (seperator, string, [limit]);
split (regular expression, string);
substr (string, start position, [length]);
```

PHP String:Join&Split (cont.)

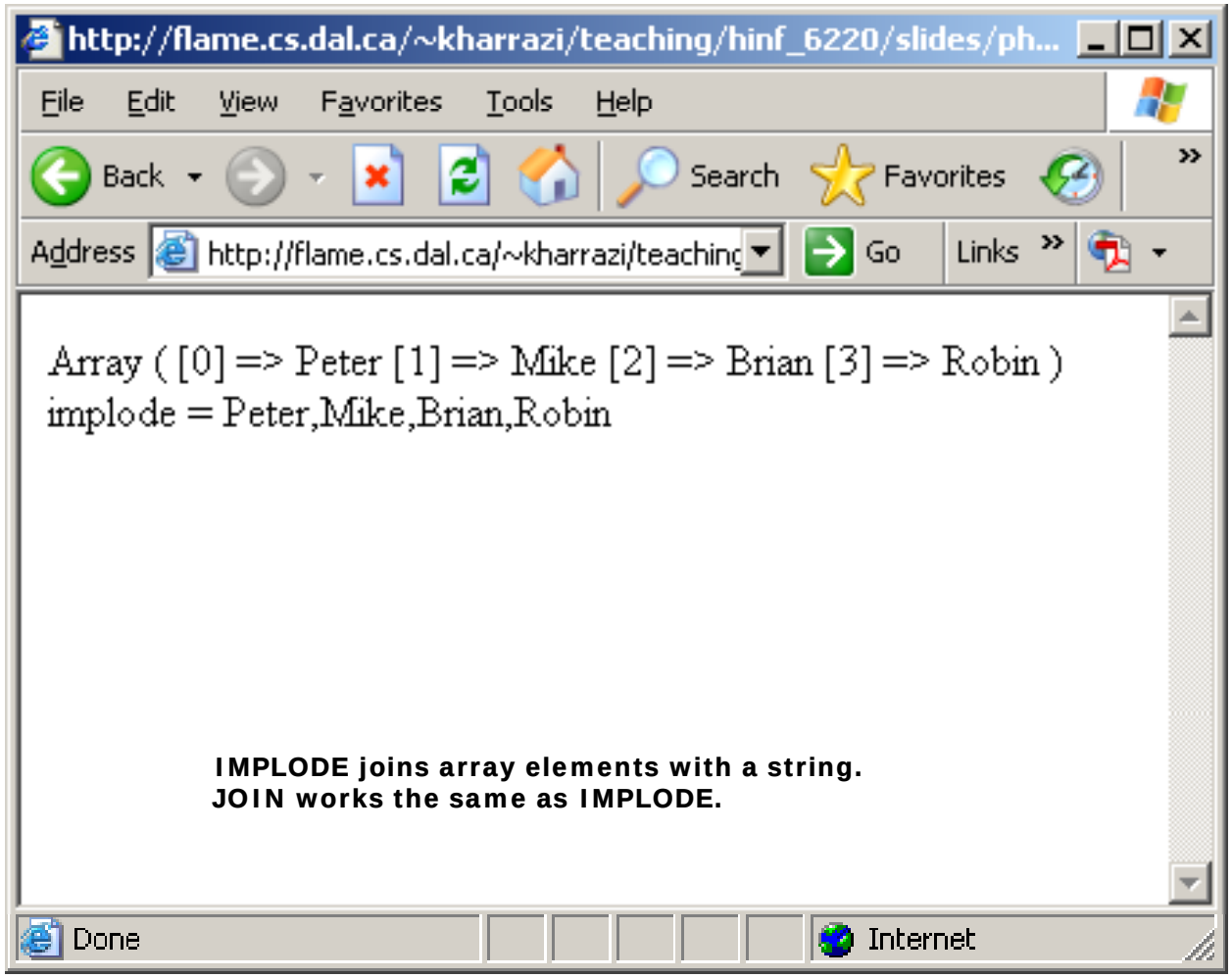
implode

```
<?php
    $text = array ("Peter", "Mike", "Brian", "Robin");

    print_r($text);

    // implode joins array elements with a string
    echo "<br>";
    echo "implode = " . implode(",", $text);
?>
```

In this case “,” (comma) is the *delimiter* or the *glue* to join the array elements.

PHP String:Join&Split (cont.)

implode



PHP String:Join&Split (cont.)

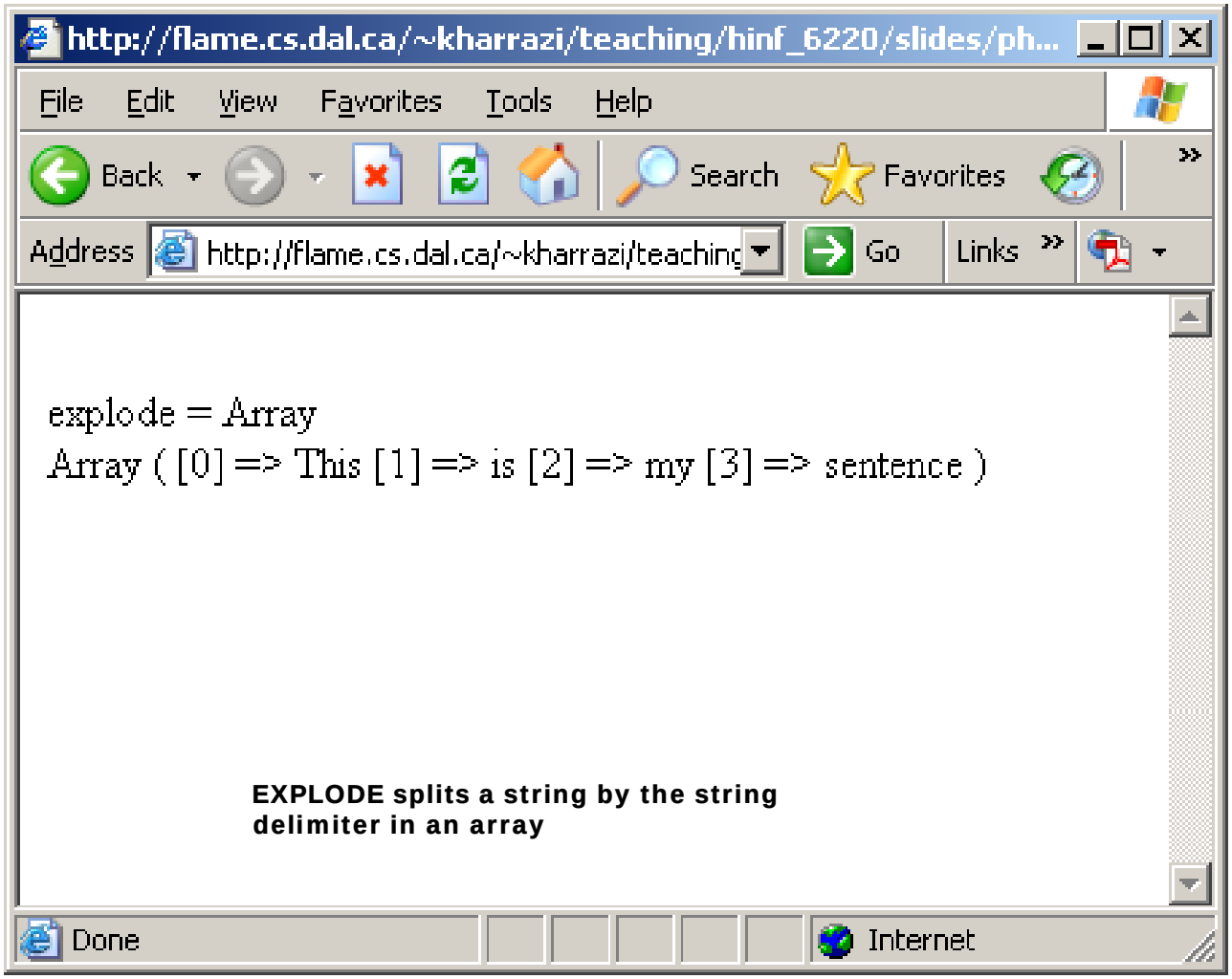
explode

```
<?php
    $text = "This is my sentence";

    // explode splits a string by the string delimiter in
    an array
    echo "<br>";
    echo "explode = " . explode(" ", $text);
    echo "<br>";
    $x = explode(" ", $text);
    print_r($x);

?>
```

In this case " " (space) is the *delimiter* or the *splitter* to split the string into fragments of then putting them as array elements.

PHP String:Join&Split (cont.)

explode



PHP String:Join&Split (cont.)

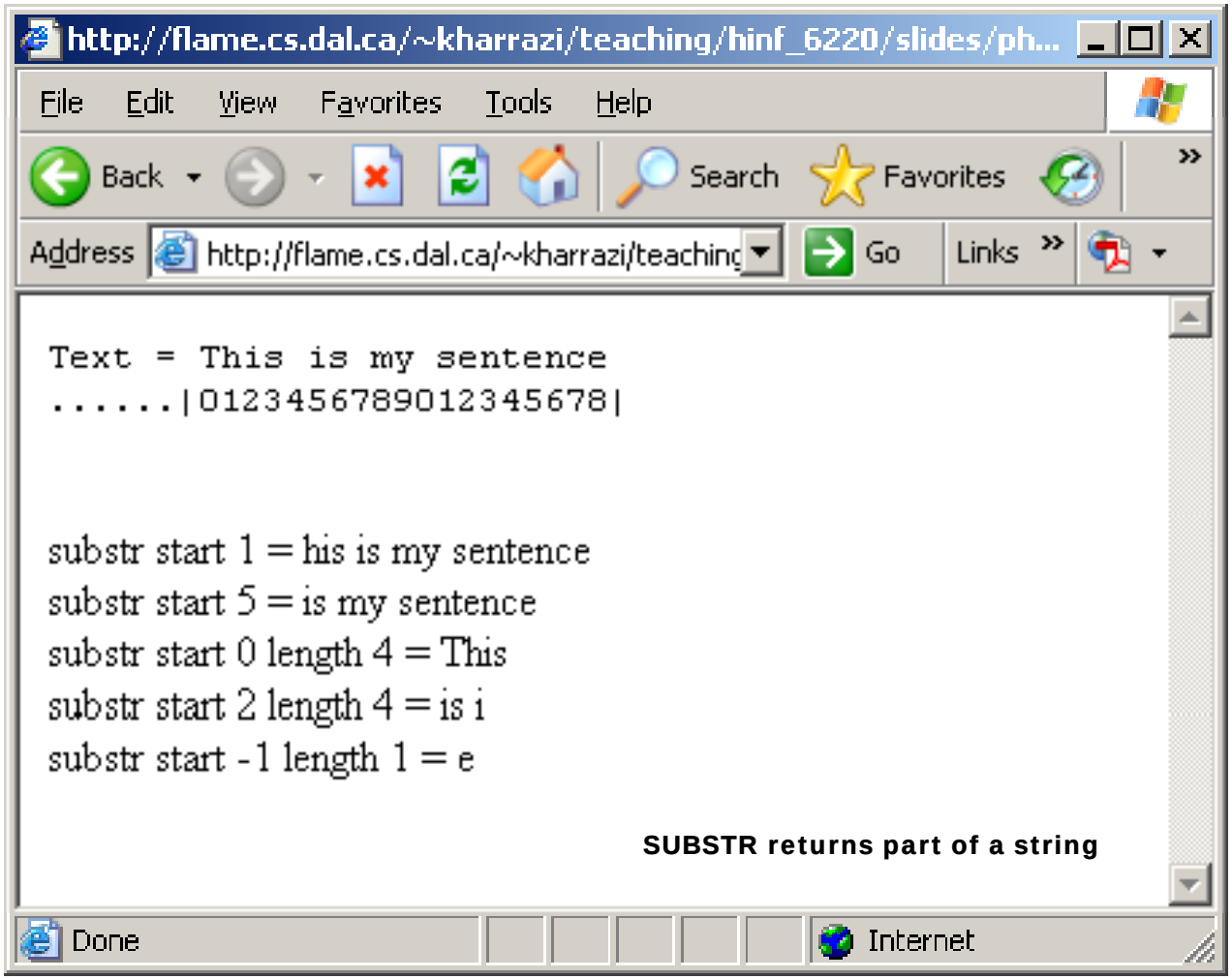
substr

```
<?php
      $text = "This is my sentence";

      echo "<pre>"; echo "Text = " . $text; echo "<br>";
      echo "......|0123456789012345678|"; echo "</pre>";

      // substr returns part of a string
      echo "<br>"; echo "substr start 1 = " . substr($text, 1);
      echo "<br>"; echo "substr start 5 = " . substr($text, 5);
      echo "<br>"; echo "substr start 0 length 4 = " . substr($text, 0, 4);
      echo "<br>"; echo "substr start 2 length 4 = " . substr($text, 2, 4);
      echo "<br>"; echo "substr start -1 length 1 = " . substr($text, -1, 1);

?>
```

substr (string, num1, num2): The first number is the starting position in the string and the second number is the length of the substring.

PHP String:Join&Split (cont.)

substr



PHP String:Comparision

- strcmp() will compare two given strings and returns < 0 if str1 is less than str2; > 0 if str1 is greater than str2, and 0 if they are equal.

- strcasecmp() is the non-case sensitive version of it.

```
strcmp (string_1, string_2);
strcasecmp (string_1, string_2);
```

- strlen() returns the length of a string

```
strlen (string);
```

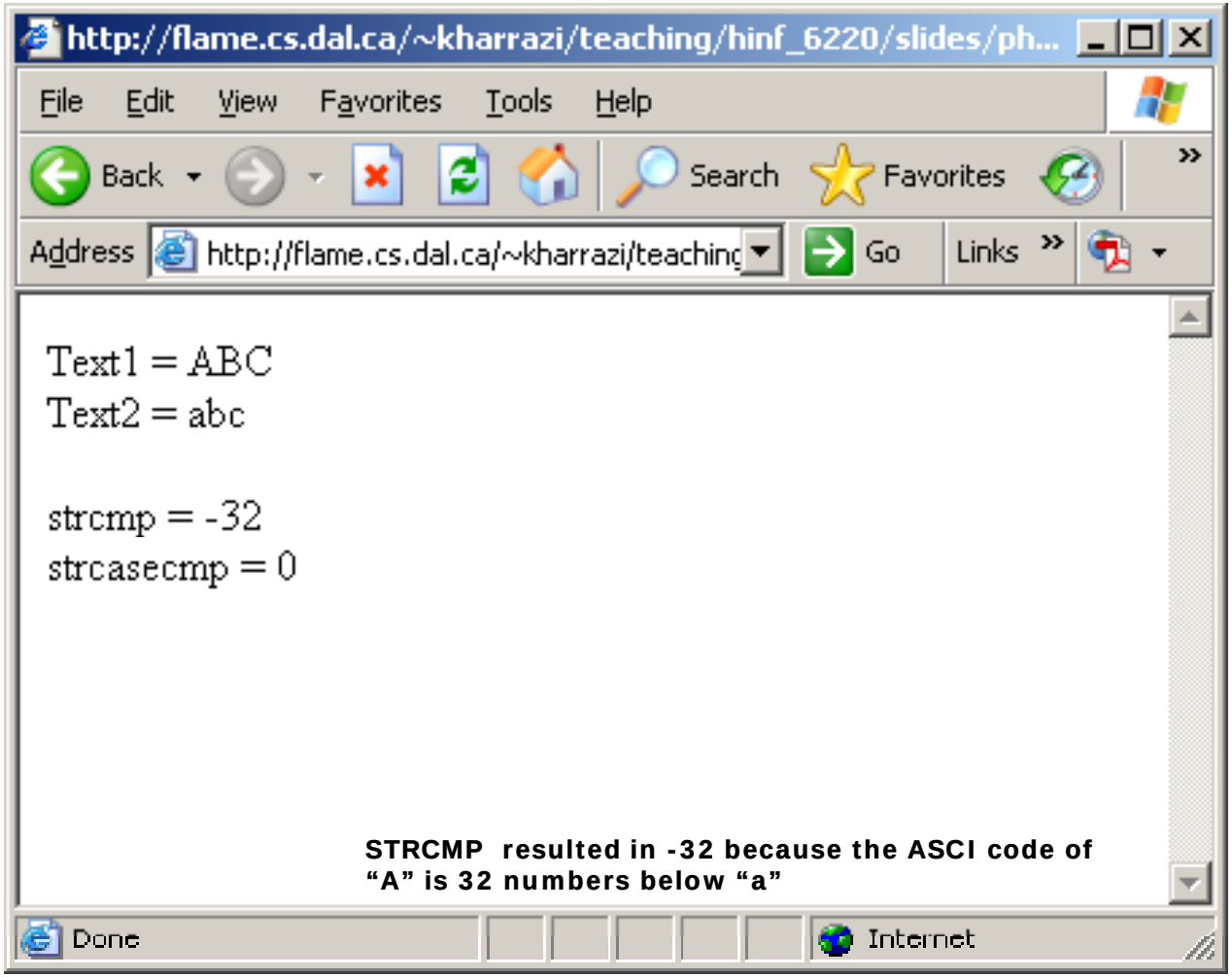
PHP String:Comparision (cont.)

strcmp
strcasecmp

```php
<?php
    $text1 = "ABC";  $text2 = "abc";
    echo "Text1 = " . $text1; echo "<br>";
    echo "Text2 = " . $text2; echo "<br>";

    // strcmp is a binary safe comparision
    echo "<br>";
    echo "strcmp = " . strcmp($text1, $text2);

    // strcasecmp is a binary safe case insensitive comparision
    echo "<br>";
    echo "strcasecmp = " . strcasecmp($text1, $text1);
?>
```

PHP String:Comparision (cont.)

strcmp
strcasecmp



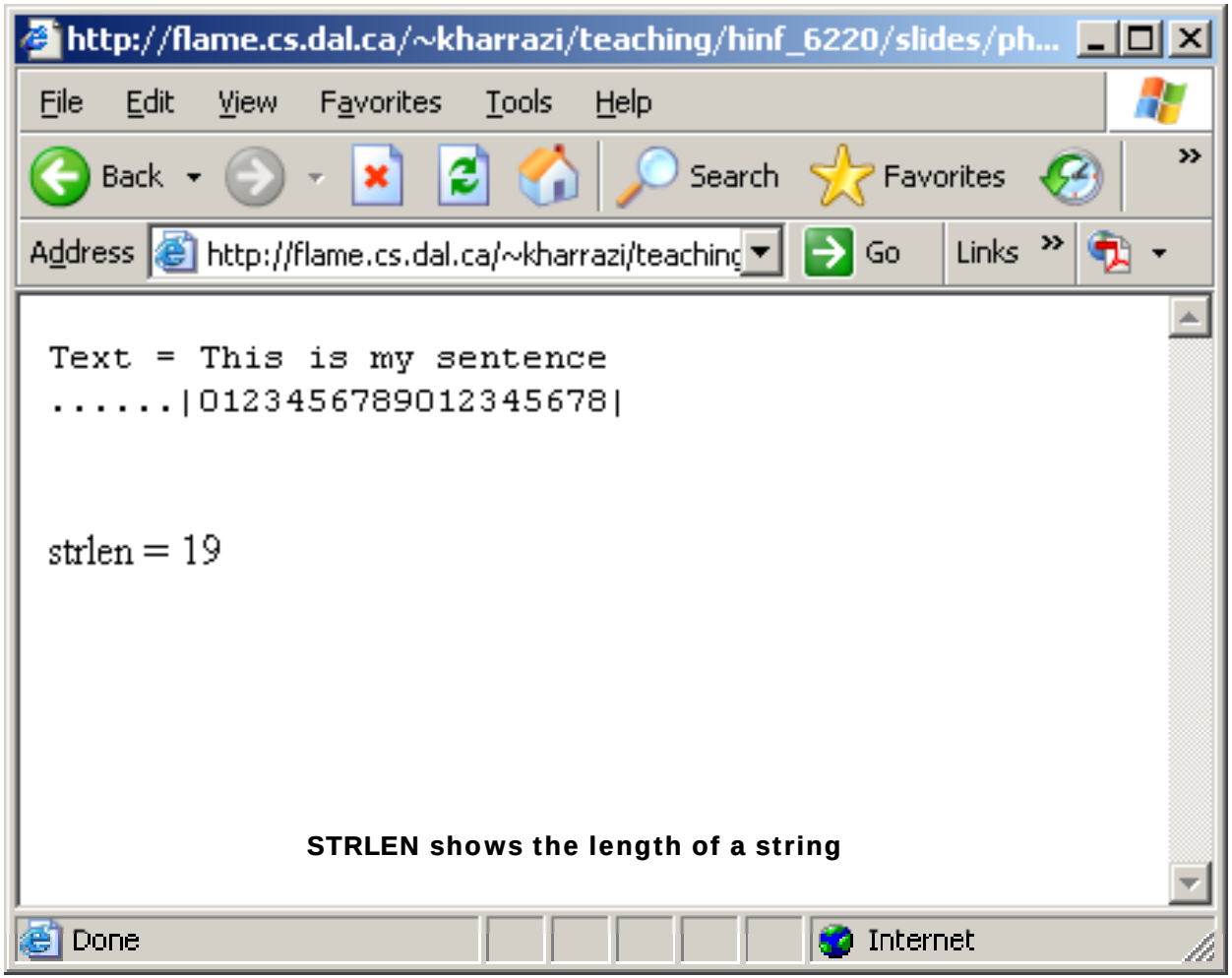
PHP String:Comparision (cont.)

strlen

```
<?php
    $text = "This is my sentence";

    echo "<pre>";
    echo "Text = " . $text;
    echo "<br>";
    echo "......|01234567890123456789|";
    echo "</pre>";

    // strlen gets the string's length
    echo "<br>";
    echo "strlen = " . strlen($text);
?>
```

PHP String:Comparision (cont.)

strlen



PHP String:Match&Replace

- Match: The commands match a specific substring in a longer string.

```
strstr (string, substring);
strchr (string, substring);
strrchr (string, substring);
stristr (string, substring);
```

- Replace: The commands replace a substring of a longer string.

```
str_replace (sub_string, sub_new, string);
```

PHP String:Match&Replace (cont.)

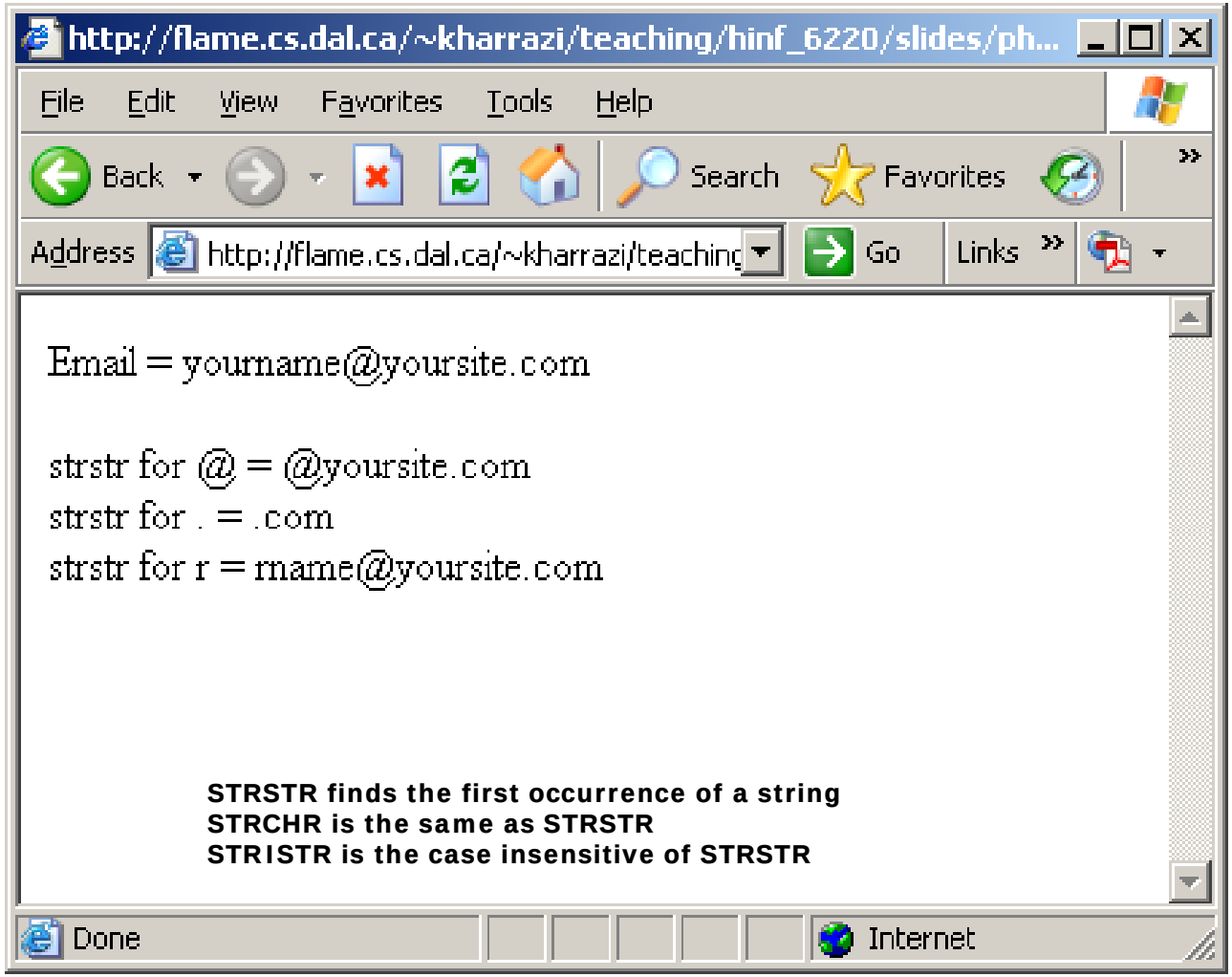
strstr

```
<?php
    $email = "yourname@yoursite.com";
    echo "Email = " . $email; echo "<br>";

    // strstr finds the first occurrence of a string
    echo "<br>";
    echo "strstr for @ = " . strstr($email, "@");
    echo "<br>";
    echo "strstr for . = " . strstr($email, ".");
    echo "<br>";
    echo "strstr for r = " . strstr($email, "r");
?>
```

In these case @, . and r have been used as the determining characters.

PHP String:Match&Replace (cont.)

strstr



PHP String:Match&Replace (cont.)

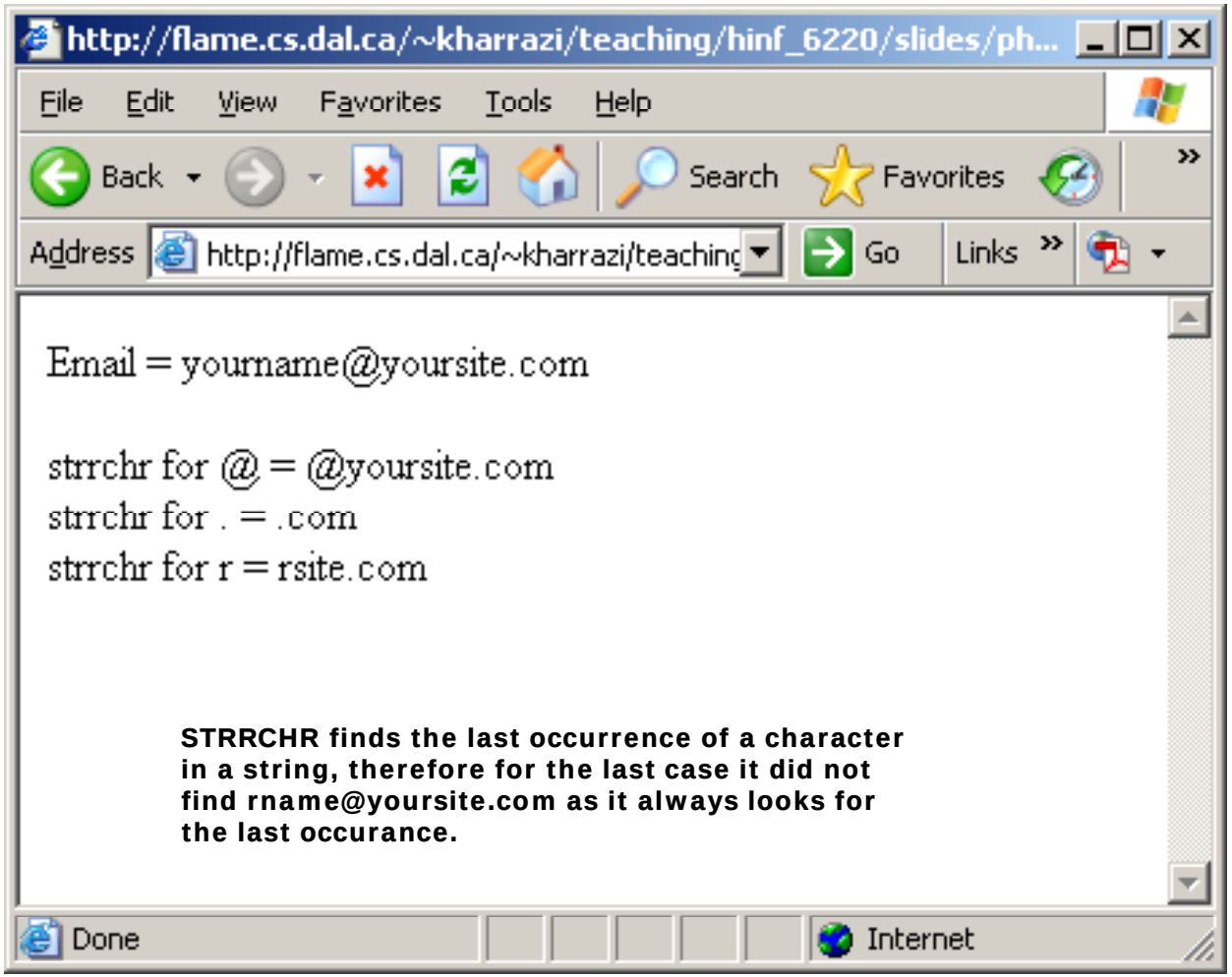
strrchr

```
<?php
    $email = "yourname@yoursite.com";
    echo "Email = " . $email; echo "<br>";

    // strrchr finds the last occurrence of a character in a string
    echo "<br>";
    echo "strrchr for @ = " . strrchr($email, "@");
    echo "<br>";
    echo "strrchr for . = " . strrchr($email, ".");
    echo "<br>";
    echo "strrchr for r = " . strrchr($email, "r");
?>
```

In these case @, . and r have been used as the determining characters.

PHP String:Match&Replace (cont.)

strrchr



PHP String:Match&Replace (cont.)

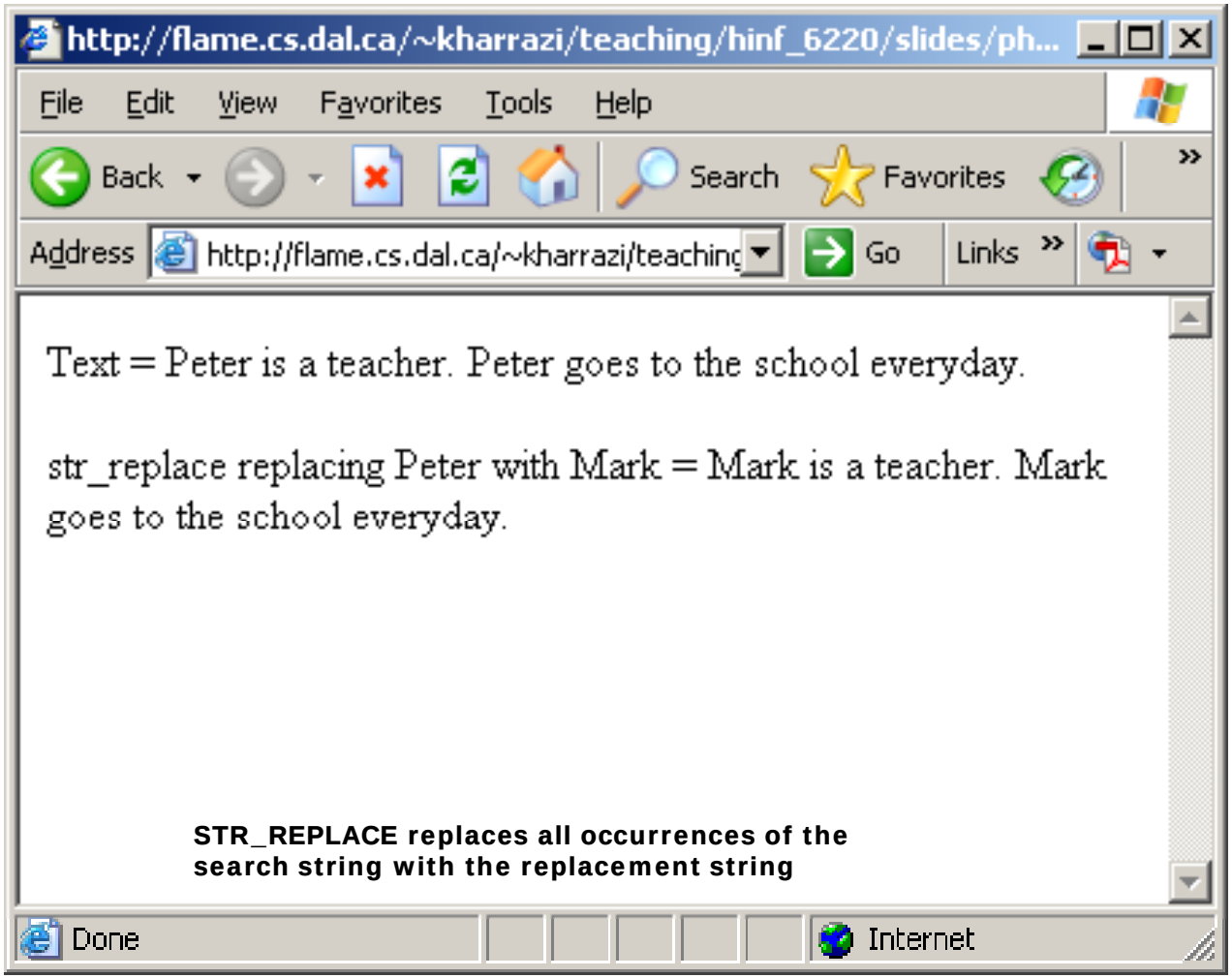
str_replace

```
<?php
    $text = "Peter is a teacher. Peter goes to the school
    everyday.";
    echo "Text = " . $text;  echo "<br>";

    // str_replace replaces all occurrences of the
    // search string with the replacement string
    echo "<br>";
    echo "str_replace replacing Peter with Mark = " .
    str_replace("Peter", "Mark", $text);
?>
```

In these case “Peter” has been replaced with “Mark” anywhere in the string.

PHP String:Match&Replace (cont.)

str_replace



2. PHP Array Manipulation

- These functions allow you to interact with and manipulate arrays in various ways. Arrays are essential for storing, managing, and operating on sets of variables.

- Simple and multi-dimensional arrays are supported, and may be either user created or created by another function. There are specific database handling functions for populating arrays from database queries, and several functions return arrays.

- For array operators please refer to: http://ca.php.net/manual/en/language.operators.array.php

PHP Array Manipulation(cont.)

- Sorting Arrays
- Reordering Arrays
- Counting Arrays
- Miscellaneous Function
- Looping Arrays
  - *(Would be covered in the LOOPS session)*

PHP Array:Sorting (cont.)

- Usually we use these functions to sort the elements of an array in order to present them in an order. Sorting could be applied on the elements and/or the indexes of the arrays.

- [] syntax means that is optional.

```
sort (array, [flag]);
asort (array, [flag]);
ksort (array, [flag]);
rsort (array, [flag]);
arsort (array, [flag]);
krsort (array, [flag]);
```

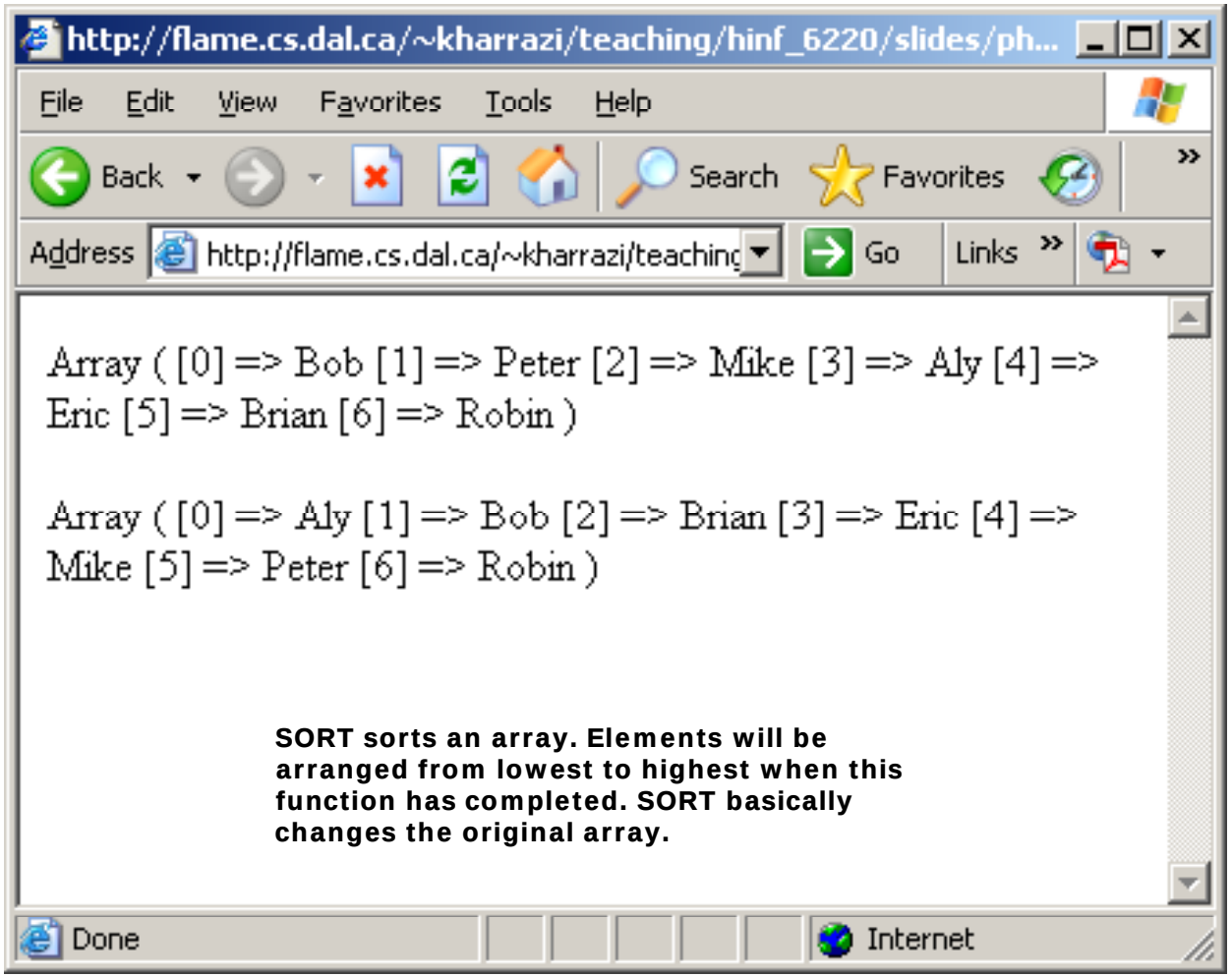
PHP Array:Sorting (cont.)

sort

```
<?php
    $x = array ("Bob", "Peter", "Mike", "Aly", "Eric",
    "Brian", "Robin");

    print_r($x);
    echo "<br>";

    //  SORT sorts an array. Elements will be arranged from
    // lowest to highest when this function has completed.
    sort ($x);
    echo "<br>";
    print_r($x);
?>
```

PHP Array:Sorting (cont.)

sort



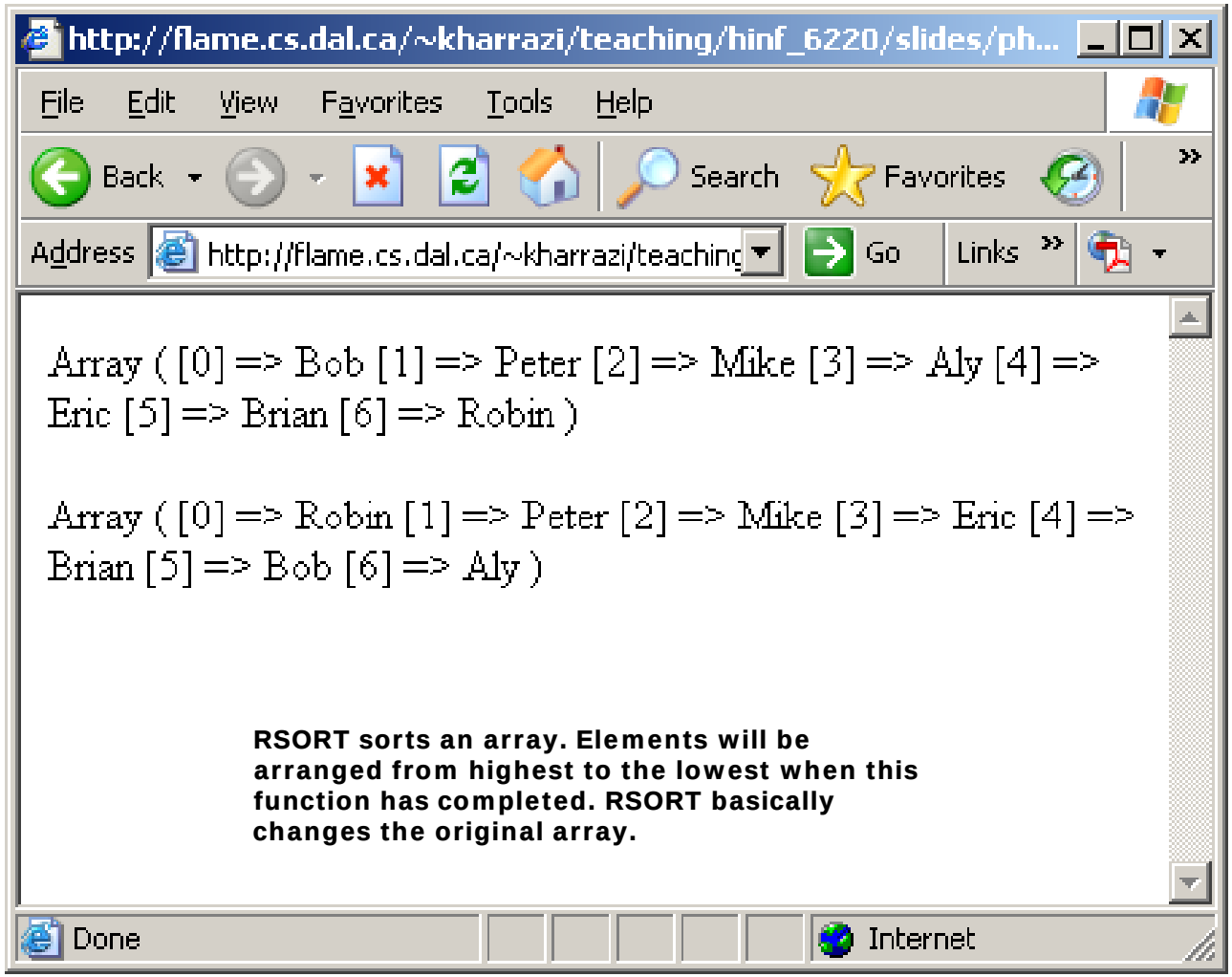
PHP Array:Sorting (cont.)

rsort

```
<?php
    $x = array ("Bob", "Peter", "Mike", "Aly", "Eric",
    "Brian", "Robin");

    print_r($x);
    echo "<br>";

    // RSORT sorts an array. Elements will be arranged from
    // highest to lowest when this function has completed.
    rsort ($x);
    echo "<br>";
    print_r($x);
?>
```

PHP Array:Sorting (cont.)

rsort



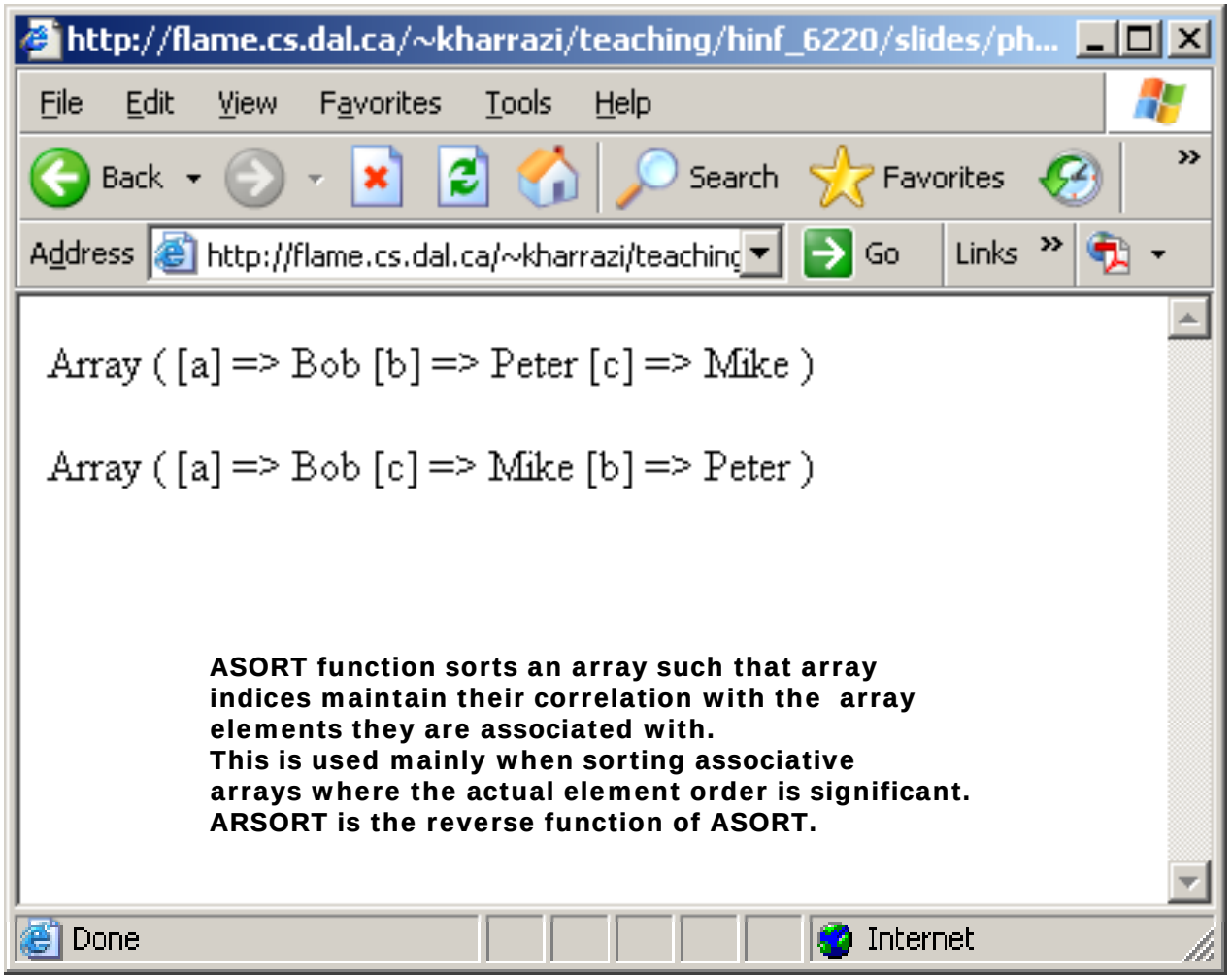
PHP Array:Sorting (cont.)

asort

```
<?php
    $x = array ("a" => "Bob", "b" => "Peter", "c" =>
    "Mike");
    print_r($x);
    echo "<br>";

    // ASORT function sorts an array such that array
    // indices maintain their correlation with the
    // array elements they are associated with.
    asort ($x);
    echo "<br>";
    print_r($x);
?>
```

PHP Array:Sorting (cont.)

asort



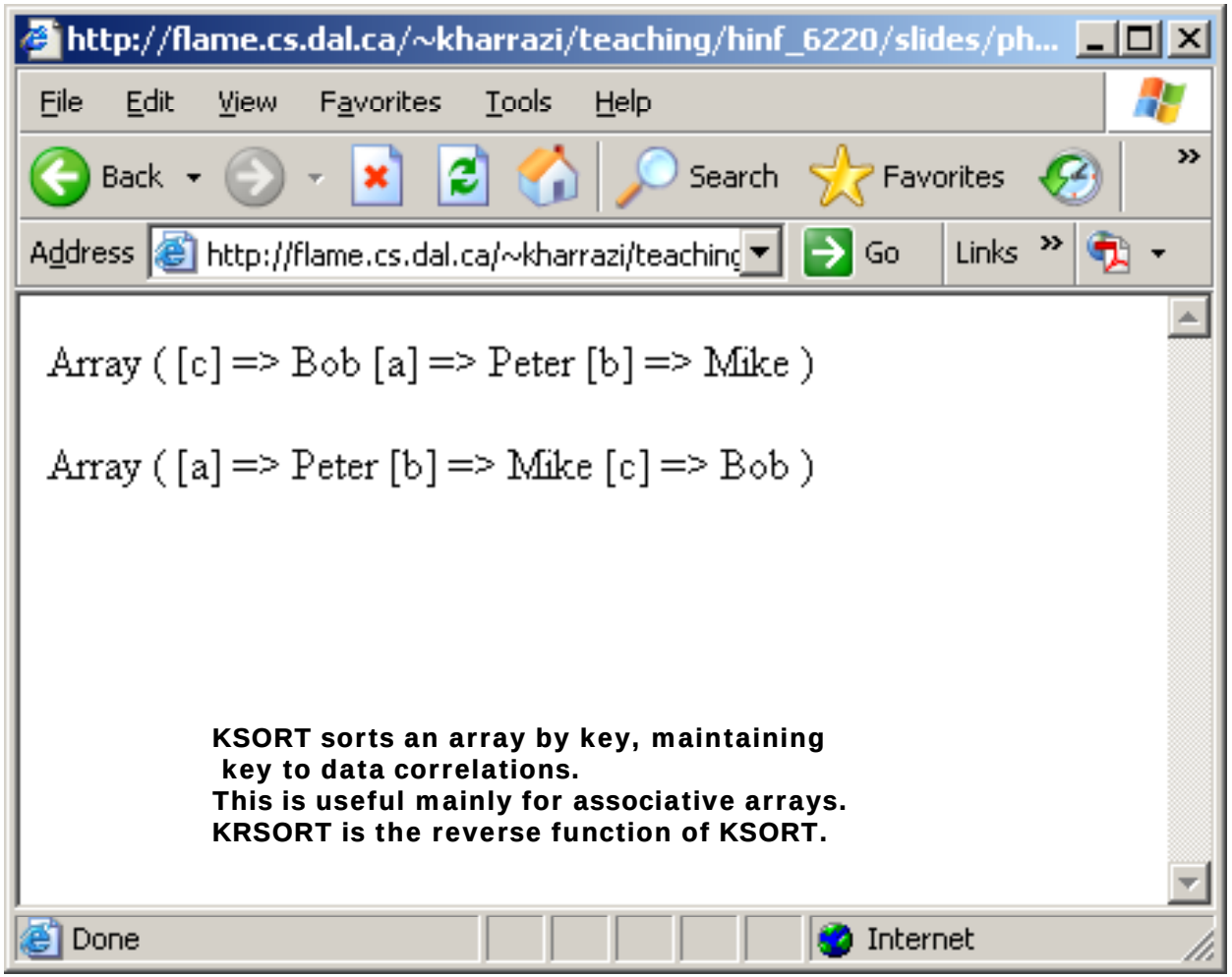
PHP Array:Sorting (cont.)

ksort

```
<?php
    $x = array ("c" => "Bob", "a" => "Peter", "b" =>
    "Mike");
    print_r($x);
    echo "<br>";

    // KSORT sorts an array by key, maintaining
    // key to data correlations. This is useful
    // mainly for associative arrays.
    ksort ($x);
    echo "<br>";
    print_r($x);
?>
```

PHP Array:Sorting (cont.)

ksort



PHP Array:Reordering (cont.)

- Usually we use these functions to reorder the elements existing in an array or to add/delete an element for the array. Some of these functions are similar to the functions mentioned in the string manipulations.

- [] syntax means that is optional.

```
array_reverse (array);
array_pop (array);
array_shift (array);
array_unshift (array, new elements);
array_push (array, new elements);
array_slice (array, start pos, [length]);
array_splice (array, start pos, [length], [new items])
```

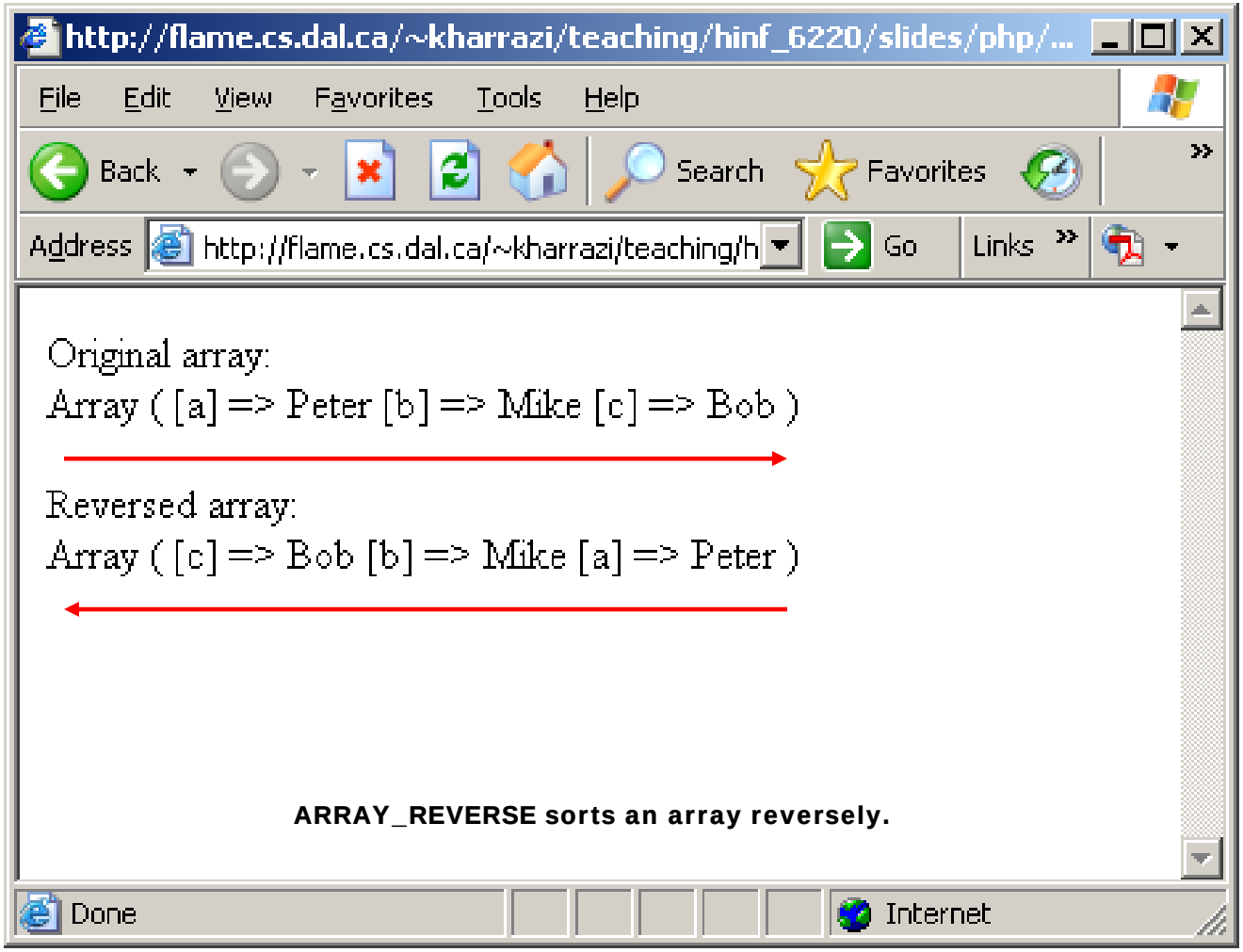
PHP Array:Reordering (cont.)

array_reverse

```
<?php
    $x = array ("a" => "Peter", "b" => "Mike", "c" =>
    "Bob");
    echo "Original array: <br>";
    print_r($x);
    echo "<br>";

    // ARRAY_REVERSE returns an array with elements in reverse order
    $x = array_reverse ($x);
    echo "<br>";
    echo "Reversed array: <br>";
    print_r($x);
?>
```

PHP Array:Reordering (cont.)

array_reverse



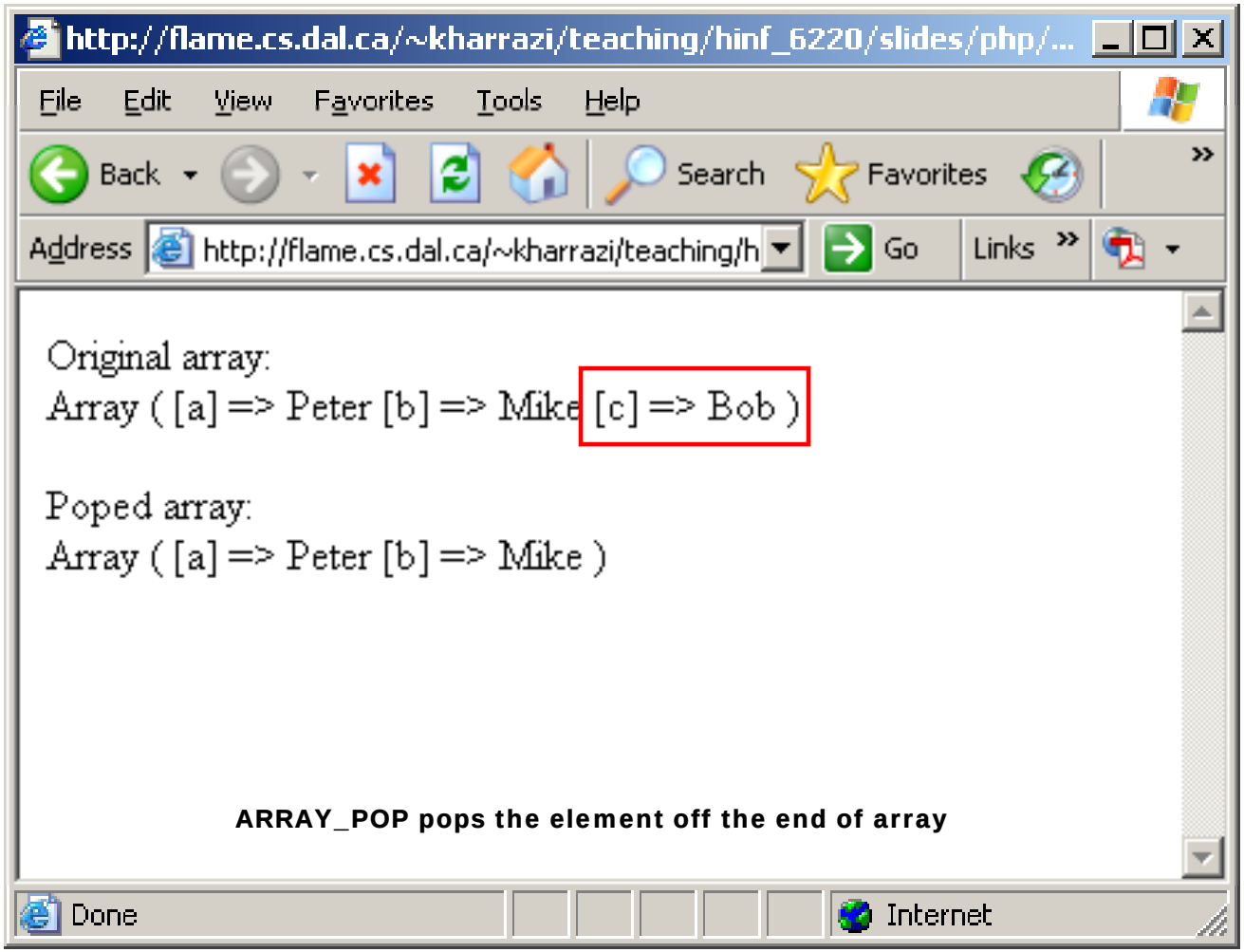
PHP Array:Reordering (cont.)

array_pop

```php
<?php
    $x = array ("a" => "Peter", "b" => "Mike", "c" =>
    "Bob");
    echo "Original array: <br>";
    print_r($x);
    echo "<br>";

    // ARRAY_POP pops the last element of the array
    array_pop ($x);
    echo "<br>";
    echo "Poped array: <br>";
    print_r($x);
?>
```

PHP Array:Reordering (cont.)

array_pop



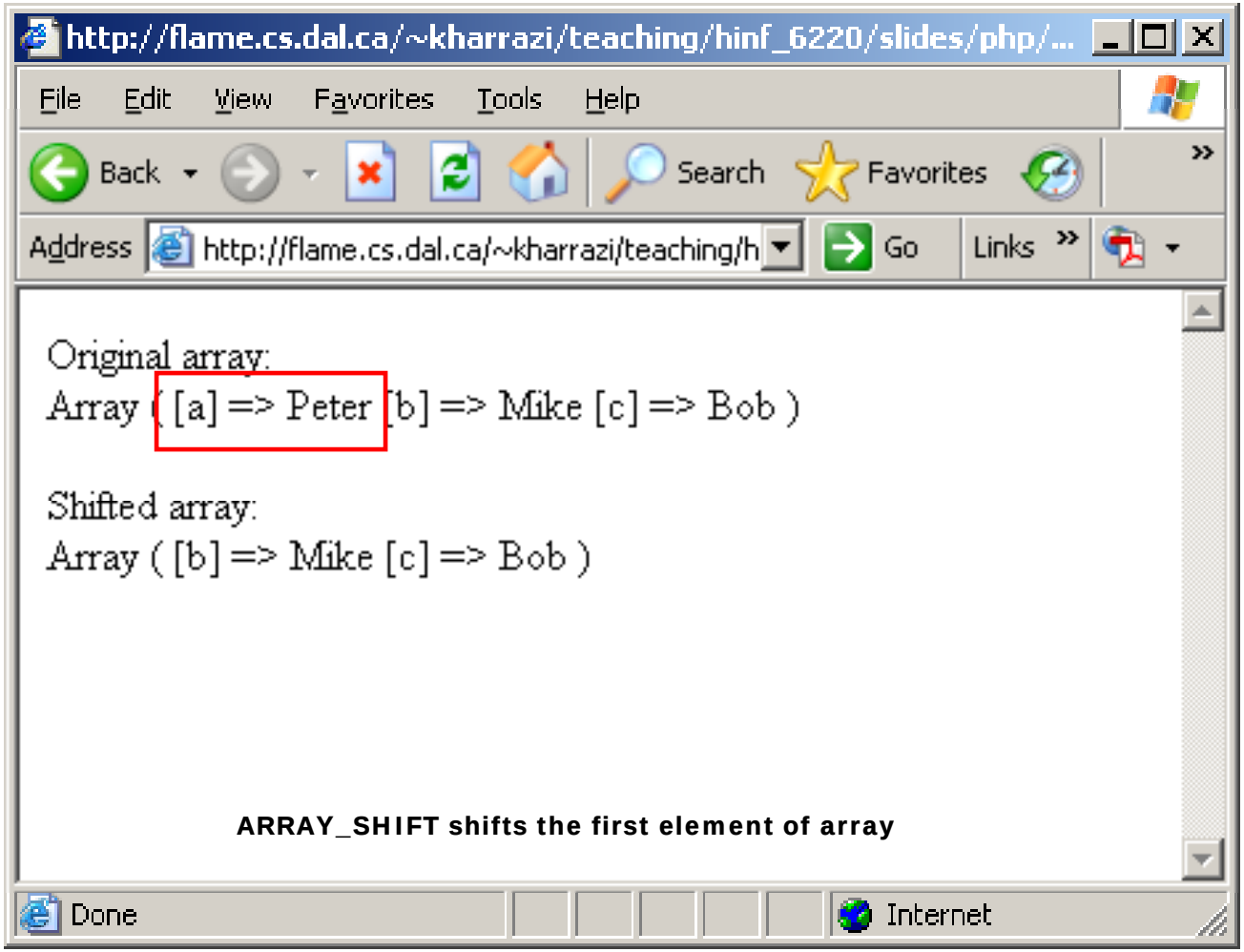
PHP Array:Reordering (cont.)

array_shift

```
<?php
    $x = array ("a" => "Peter", "b" => "Mike", "c" =>
    "Bob");
    echo "Original array: <br>";
    print_r($x);
    echo "<br>";

    // ARRAY_SHIFT shifts the first element of the array
    array_shift ($x);
    echo "<br>";
    echo "Shifted array: <br>";
    print_r($x);
?>
```

PHP Array:Reordering (cont.)

array_shift



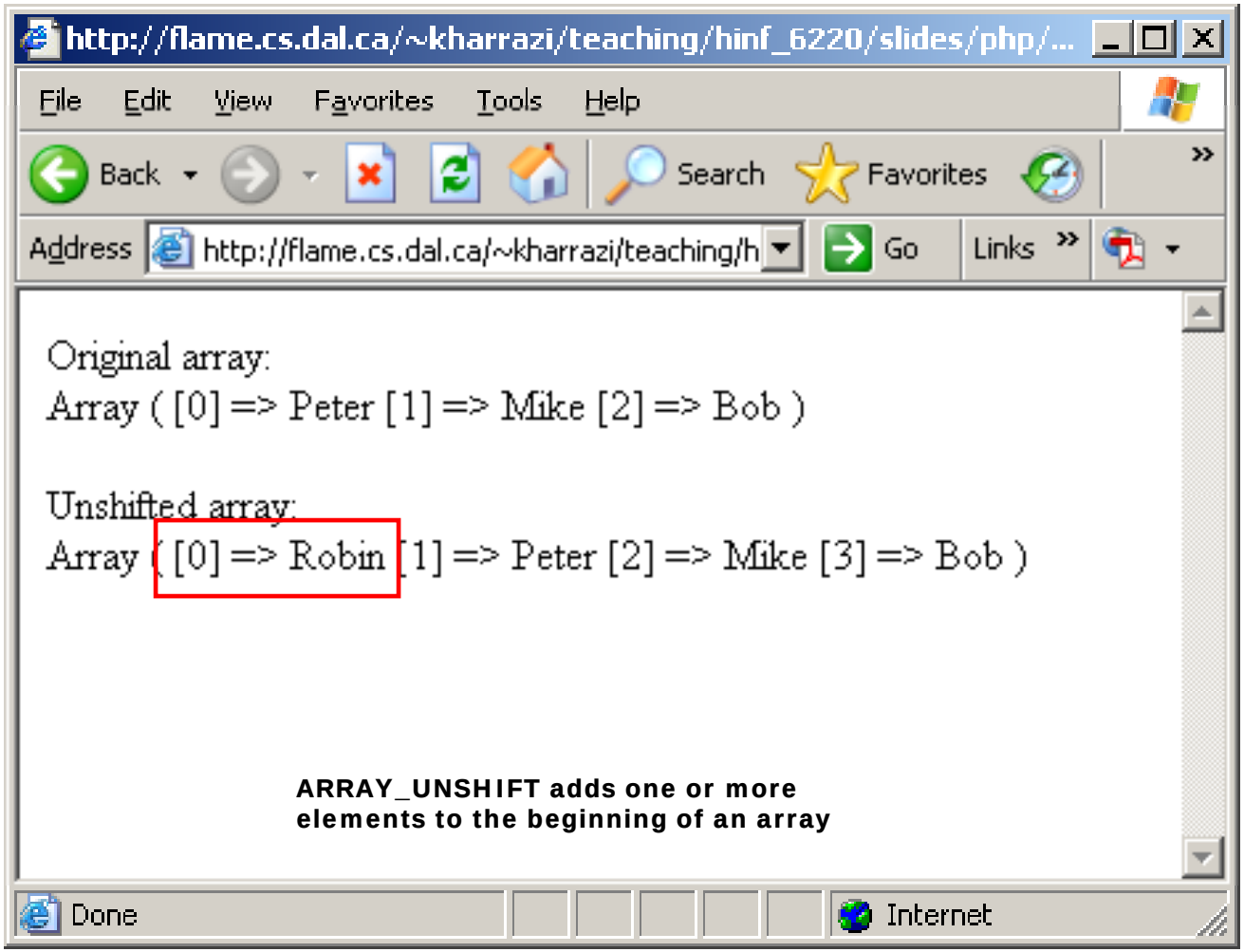
PHP Array:Reordering (cont.)

array_unshift

```
<?php
     $x = array ("Peter", "Mike", "Bob");
     echo "Original array: <br>";
     print_r($x);
     echo "<br>";

     // ARRAY_UNSHIFT adds an element to the beginning of the array
     array_unshift ($x, "Robin");
     echo "<br>";
     echo "Unshift array: <br>";
     print_r($x);
?>
```

PHP Array:Reordering (cont.)

array_unshift



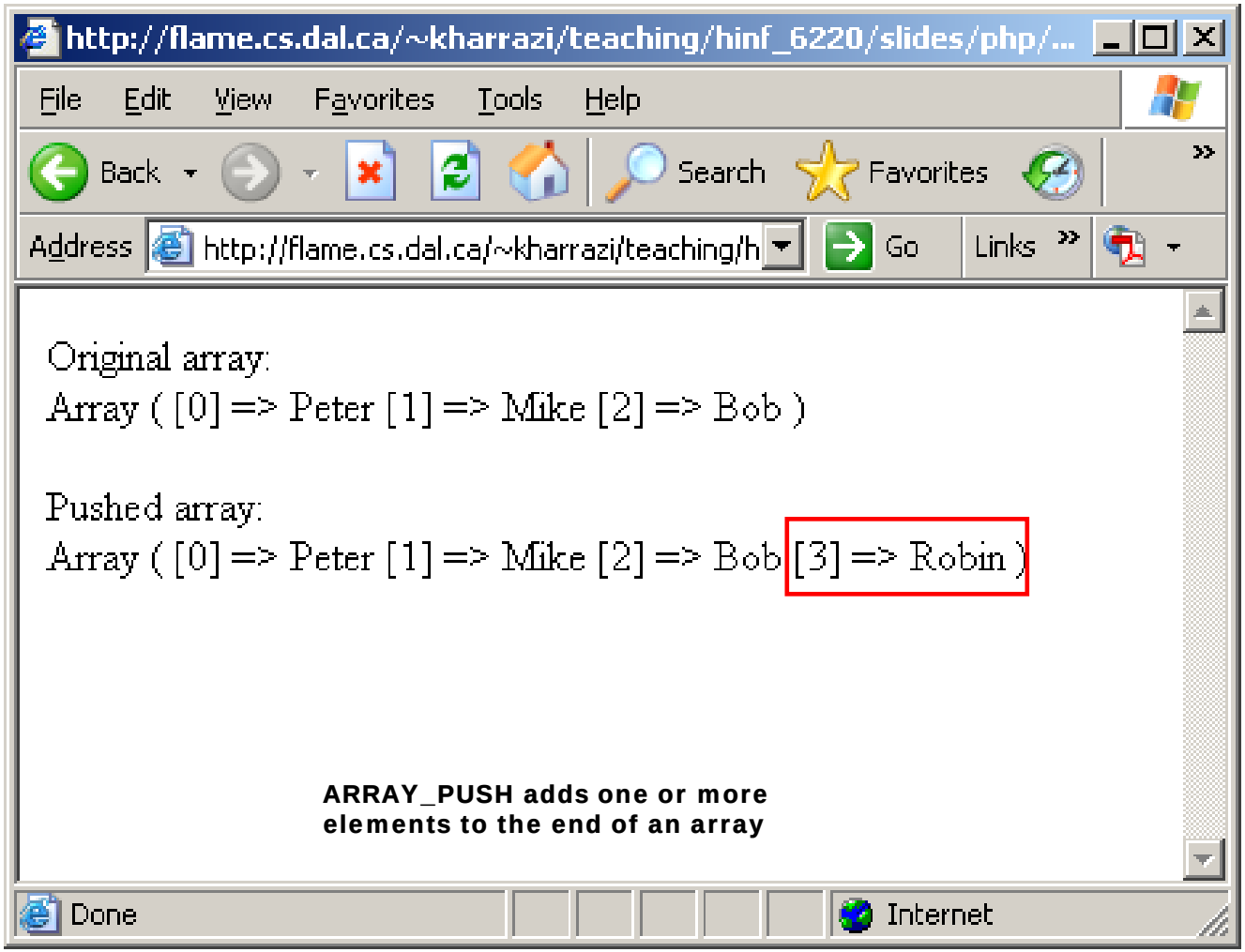
PHP Array:Reordering (cont.)

array_push

```
<?php
     $x = array ("Peter", "Mike", "Bob");
     echo "Original array: <br>";
     print_r($x);
     echo "<br>";

     // ARRAY_PUSH adds an element to the end of the array
     array_push ($x, "Robin");
     echo "<br>";
     echo "Pushed array: <br>";
     print_r($x);
?>
```

PHP Array:Reordering (cont.)

array_push



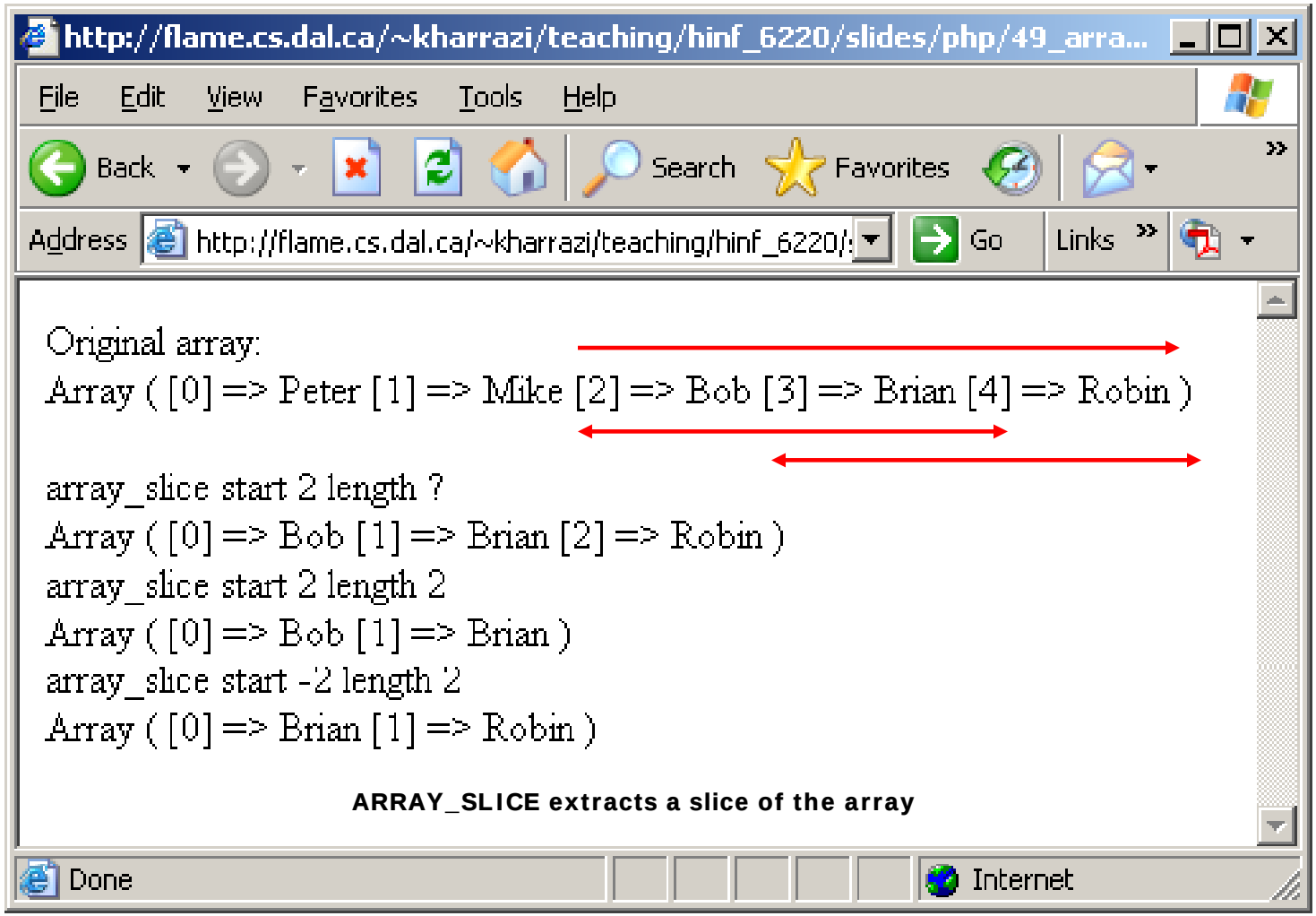
PHP Array:Reordering (cont.)

array_slice

```
<?php
    $x = array ("Peter", "Mike", "Bob", "Brian", "Robin");
    echo "Original array: <br>";
    print_r($x);
    echo "<br>";

    // ARRAY_SLICE extracts a slice of the array
    echo "<br>array_slice start 2 length ? <br>";
    print_r(array_slice ($x, 2));
    echo "<br>array_slice start 2 length 2  <br>";
    print_r(array_slice ($x, 2, 2));
    echo "<br>array_slice start -2 length 2  <br>";
    print_r(array_slice ($x, -2 , 2));
?>
```

PHP Array:Reordering (cont.)

array_slice



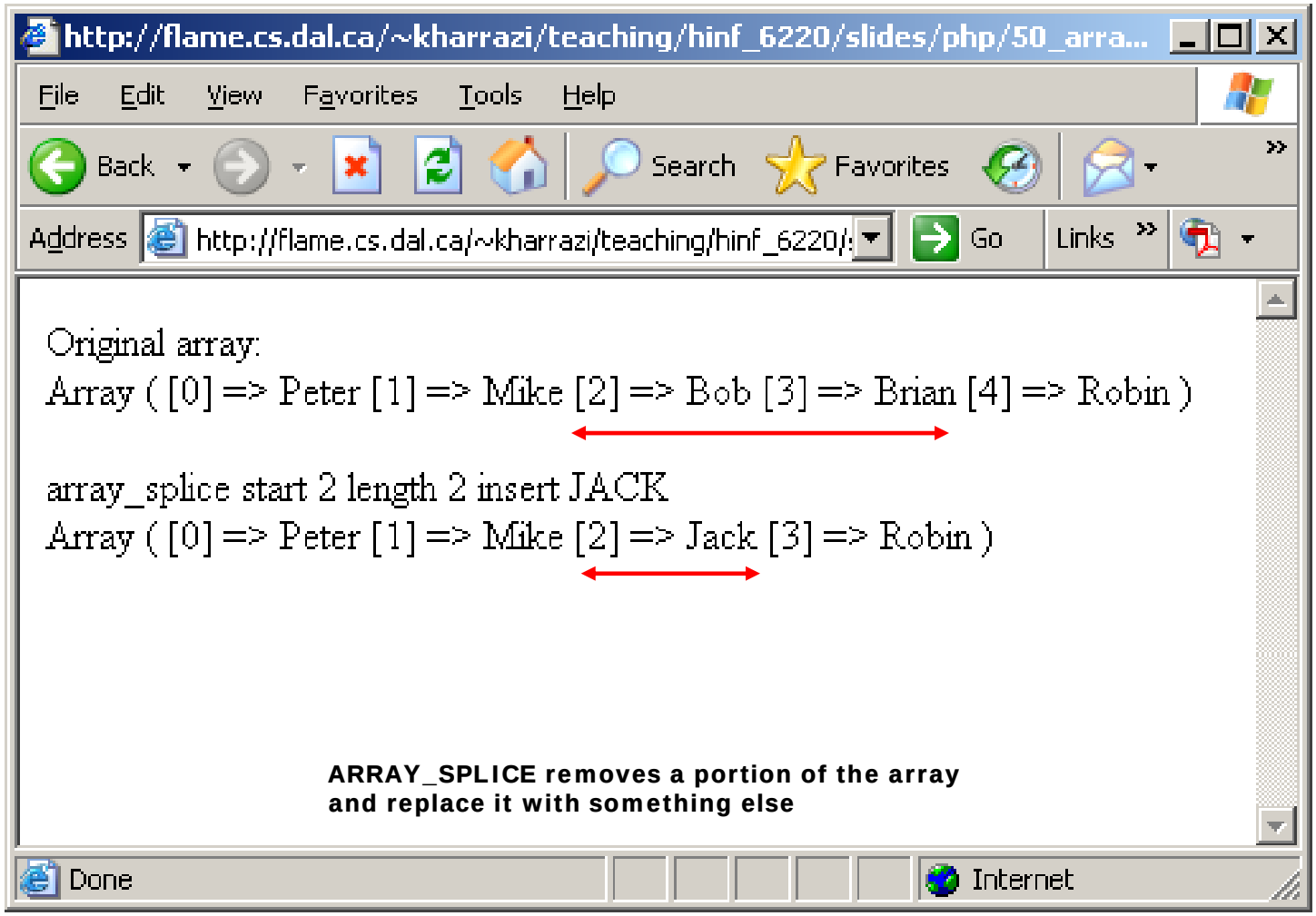
PHP Array:Reordering (cont.)

array_splice

```
<?php
    $x = array ("Peter", "Mike", "Bob", "Brian", "Robin");
    echo "Original array: <br>";
    print_r($x);
    echo "<br>";

    // ARRAY_SPLICE removes a portion of the array and
    replace it with something else
    echo "<br>array_splice start 2 length 2 insert
    JACK<br>";
    array_splice ($x, 2, 2, "Jack");
    print_r($x);
?>
```

PHP Array:Reordering (cont.)

array_splice



PHP Array:Count

- In many cases we will need to count the number of elements in an array in order to limit a process by reaching a number greater than the arrays elements.

- [] syntax means that is optional.

```
count(array);
sizeof (array);
```

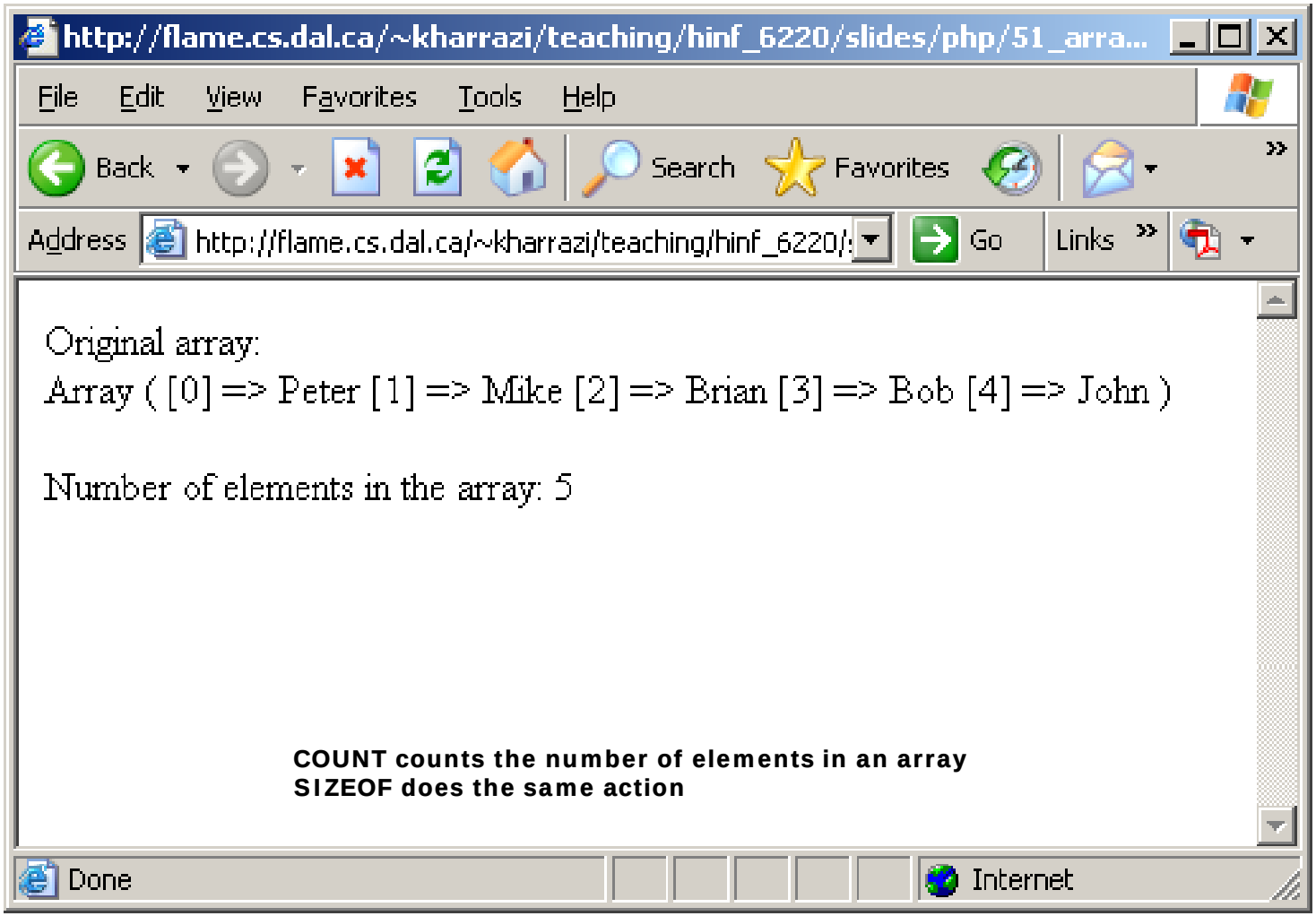
PHP Array:Count (cont.)

array_count

```
<?php
    $x = array ("Peter", "Mike", "Brian", "Bob", "John");
    echo "Original array: <br>";
    print_r($x);
    echo "<br><br>";

    // ARRAY_COUNT counts elements in an array
    echo "Number of elements in the array: " . count($x);
?>
```

PHP Array:Count (cont.)

array_count



PHP Array:Miscellaneous

- The following actions (functions) could be useful manipulating arrays specially when they are holding fetched data received from a database query.

- [] syntax means that is optional.

```
array_search (array, element);
array_merge (array1, array2);
```

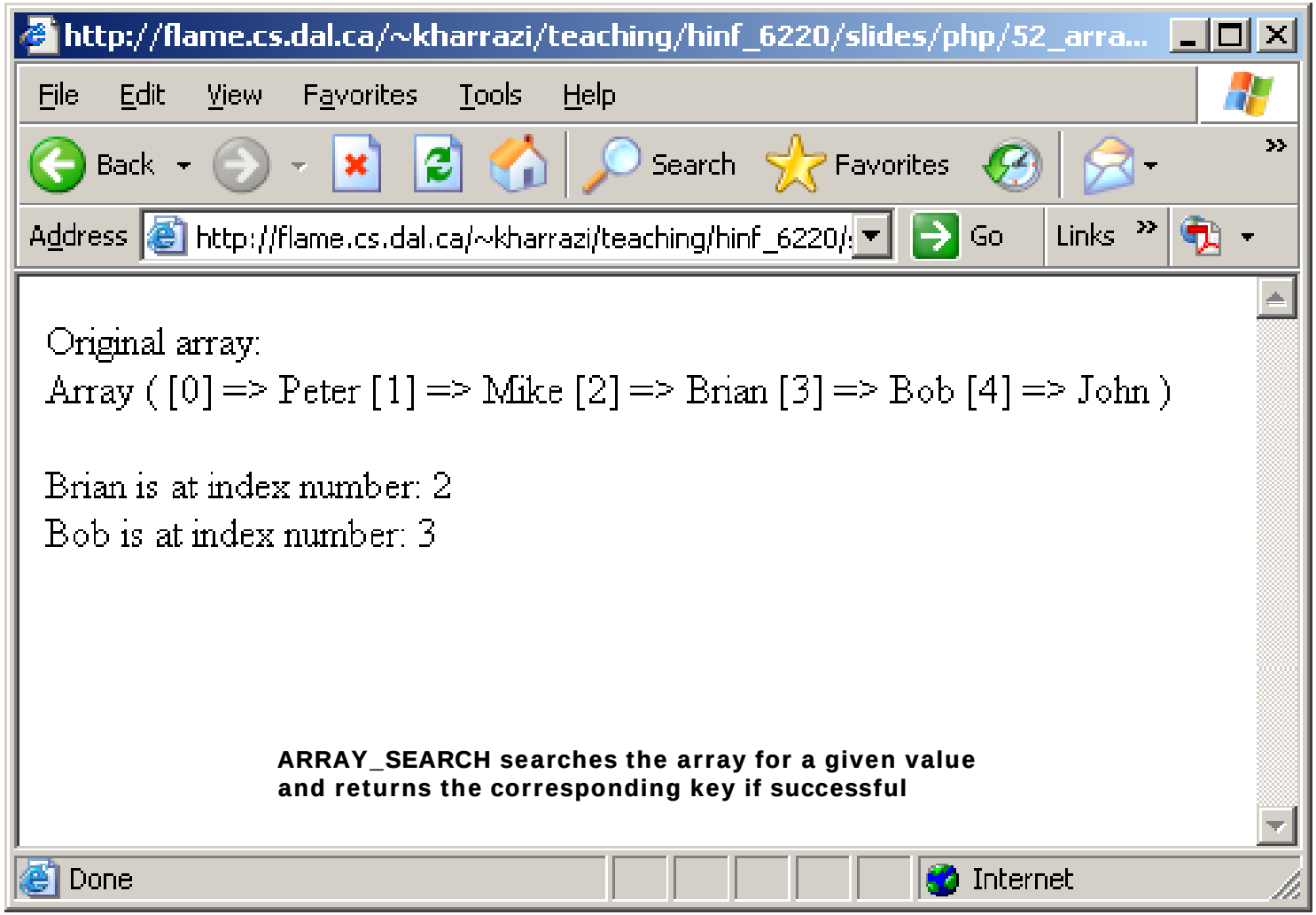
PHP Array:Miscellaneous (cont.)

array_search

```php
<?php
    $x = array ("Peter", "Mike", "Brian", "Bob", "John");
    echo "Original array: <br>";
    print_r($x);
    echo "<br><br>";

    // ARRAY_SEARCH searches the array for a given value
    and returns the corresponding key if successful

    echo "Brian is at index: " . array_search('Brian', $x);
    echo "<br>";
    echo "Bob is at index: " . array_search('Bob', $x);
?>
```

PHP Array:Miscellaneous (cont.)

array_search



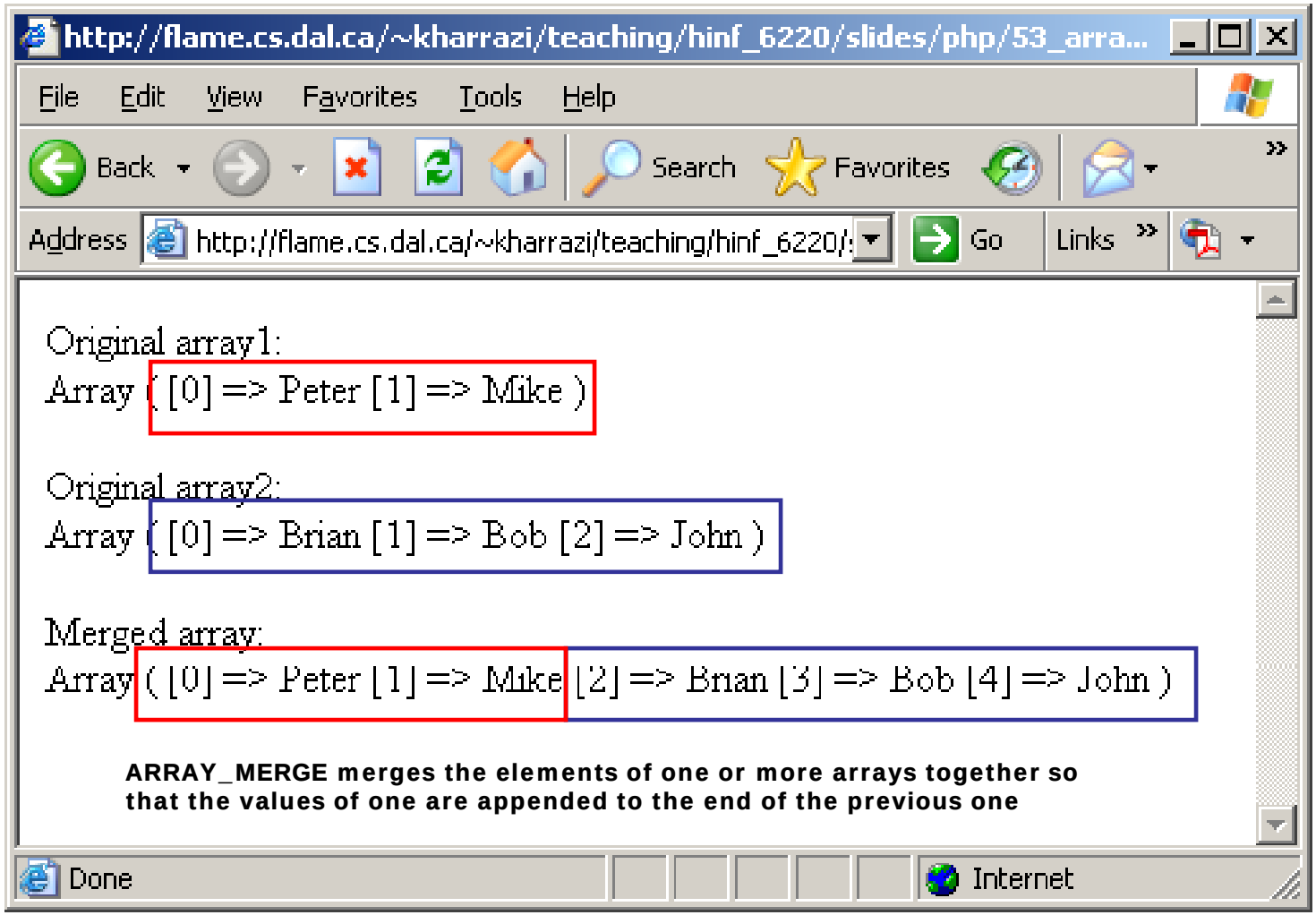
PHP Array:Miscellaneous (cont.)

array_merge

```php
<?php
    $x = array ("Peter", "Mike");
    echo "Original array1: <br>";
    print_r($x);
    echo "<br><br>";
    $y = array ("Brian", "Bob", "John");
    echo "Original array2: <br>";
    print_r($y);
    echo "<br>";
    // ARRAY_MERGE merges the elements of one or more arrays together so
    // that the values of one are appended to the end of the previous one
    $z = array_merge ($x, $y);
    echo "<br>";
    echo "Merged array: <br>";
    print_r($z);
?>
```

PHP Array:Miscellaneous (cont.)

array_merge



Summary

1. PHP: String Manipulations
   - Formatting
   - Join & Split
   - Comparison
   - Match & Replace

2. PHP: Array Manipulations
   - Sorting
   - Reordering
   - Count
   - Miscellaneous

Next Sessions

- PHP Conditions
- PHP Loops
- PHP Functions
- PHP Cookies / Sessions
- PHP SSI
- PHP Forms
- PHP/MySQL Integration

Exercise

- Please refer to the available text file in the slides section for this session on the course website:

- http://info510.com/core/public_page.php?page_name=slides