Lab 4 – Strings

Reference: Chapter 3 of the "PHP Programming with MySQL" textbook.

PHP Strings: http://php.net/manual/en/book.strings.php

Aims:

- To be able to use various string functions and practice using control structures.

Getting Started:

Create a new folder '**lab04**' under the unit folder on the mercury server *~/cos30020/www/htdocs* folder on mercury. Save today's work in this lab04 folder.

You could also create and link an external stylesheet, to the pages, and this should be valid CSS3.

Task 1: Understanding string functions (9 marks)

Step 1:
Create a file **`strprocess.php`** that will receive an input from **`strform.php`** from Step 2 via POST method, remove all the vowels then output the resulting string. It should check if the input contains only the letters and spaces using regular expression, otherwise, it should generate an appropriate error message.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="description" content="Web application development" />
  <meta name="keywords" content="PHP" />
  <meta name="author"   content="Your Name" />
  <title>TITLE</title>
</head>
<body>
    <h1>Web Programming - Lab 4</h1>
    <?php // read the comments for hints on how to answer each item
      if (isset ($_POST["     (1)     "])){ // check if form data exists
        $str = $_POST["     (2)     "];     // obtain the form data
        $pattern = "/^[A-Za-z ]+$/";     // set regular expression pattern
        if (_______(3)________) {       // check if $str with regular expression
          $ans = "";                    // initialise variable for the answer
          $len = _____(4)_______;       // obtain length of string $str
          for ($i = 0; $i < $len; $i++) {      // checks all characters in $str
            $letter = substr ($str, (5), 1);   // extract 1 char using substr
            // check using strops, is_numeric is used as strops returns a number
            // (position) if found, and false otherwise
            if ((strpos ("AEIOUaeiou", (6)) === false){
               $ans = $ans . $letter;          // concatenate letter to answer
            }
          }
          // generate answer after all letters are checked
          echo "<p>The word with no vowels is ", $ans, ".</p>";
        } else {                        // string contains invalid characters
          echo "<p>Please enter a string containing only letters or space.</p>";
        }
      } else {                          // no input
        echo "<p>Please enter string from the input form.</p>";
      }
    ?>
</body>
</html>
```

Step 2:
Create a file **`strform.php`** that contains a form with a single text box that allows a user to enter data, and submit it to s**`strprocess.php`**.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="description" content="Web application development" />
  <meta name="keywords" content="PHP" />
  <meta name="author"   content="Your Name" />
  <title>TITLE</title>
</head>
<body>
    <h1>Web Programming Form - Lab 4</h1>
    <form action = ______(7)______ method = ____(8)_____ >
        __(9-10)_____________________________________________
        _____________________________________________
        _____________________________________________
        _____________________________________________
        _____________________________________________
    </form>
</body>
</html>
```

Test in the browser, and check that the page is valid.

Task 2: Practicing string functions (3 marks)

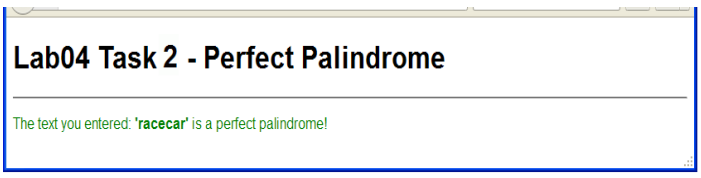
Background:
A perfect palindrome is a word or phrase that is identical forward or backward, such as the word “racecar”.
A standard palindrome is similar to a perfect palindrome except that spaces and punctuation are ignored.
For example, “Madam, I'm Adam” is a standard palindrome because the characters are identical forward or backward, provided you remove the spaces and punctuation marks.

Step 1:
Create a file **`perfectpalindromeform.php`** that contains a form with a single text box that allows a user to enter a string, and submit it to **`perfectpalindrome.php`**.

Step 2:
Create a file **`perfectpalindrome.php`** with a script that tests whether a word or phrase, entered by a user through the form is a perfect palindrome.

Hint: Use the `strrev()` function to reverse the input word or phrase and then use the `strcmp()` function to compare the original word or phrase with the reversed one.
Suggest also converting the strings to lower case, or upper case, before comparing them.

Test in the browser, and check that both webpages are valid.



Task 3: Practicing the use of `str_replace()` (3 marks)

Step 1:
Save copies of the scripts created in Task 2 as **`standardpalindromeform.php`** and **`standardpalindrome.php`**.

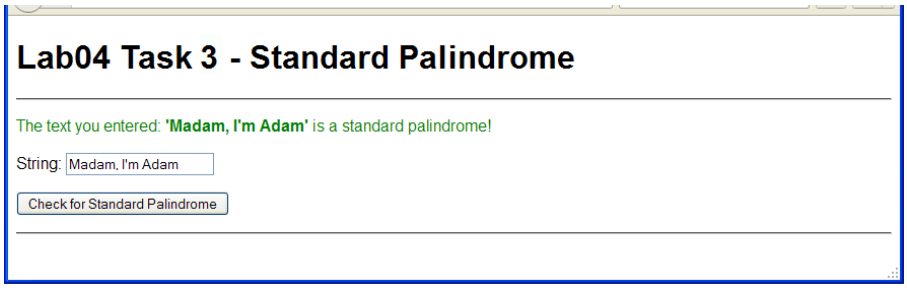
Step 2:
Modify the script to check for standard palindromes. For standard palindrome, first remove all the punctuation from the phrase using the `str_replace()` function before reversing the word or phrase and comparing it with the original one.

View in the browser, and check that both the webpages are valid.

Extra Challenge:

Combine the form **`standardpalindromeform.php`** and processing script **`standardpalindrome.php`**., incorporating the form into a script **`standardpalindromeself.php`**.. i.e. Use a single webpage with the script that displays and also processes the form, i.e. the webpage calls itself.

Test in the browser, and check that it is valid.



Note:
If you want to prevent any problems that might be caused by users including html markup, such as < or >, as input in forms, then either replace them, or use the functions `htmlspecialchars()` or `htmlentities()` to convert these characters.

See http://php.net/manual/en/function.htmlspecialchars.php
and also see http://www.php.net/manual/en/function.htmlentities.php