## Bijlage 1: Opdrachten Deel I

### B1.1 Activiteit 1

1. Create a new document in your text editor.
2. Type the <!DOCTYPE> declaration, <html> element, document <head>, and <body> element and create a script section in the document body:

<?php

?>

1. Add the following code to the script section to declare an associative array of European cities and the distances between them in kilometres. Also add a default value for the start and end index, and the conversion factor for kilometers to miles:

$distances = array(

"Berlin" => array(

"Berlin" => 0,

"Moscow" => 1607.99,

"Paris" => 876.96,

"Prague" => 280.34,

"Rome" => 1181.67

),

"Moscow" => array(

"Berlin" => 1607.99,

"Moscow" => 0,

"Paris" => 2484.92,

"Prague" => 1664.04,

"Rome" => 2374.26

),

"Paris" => array(

"Berlin" => 876.96,

"Moscow" => 641.31,

"Paris" => 0,

"Prague" => 885.38,

"Rome" => 1105.76

),

"Prague" => array(

"Berlin" => 280.34,

"Moscow" => 1664.04,

"Paris" => 885.38,

"Prague" => 0,

"Rome" => 922

),

"Rome" => array(

"Berlin" => 1181.67,

"Moscow" => 2374.26,

"Paris" => 1105.76,

"Prague" => 922,

"Rome" => 0

)

);

$startIndex = "Berlin";

$endIndex = "Berlin";

$kmToMiles = 0.62;

1. Add the following if statement to process the entered cities:

if (isset($\_POST['submit']))

{

$startIndex = stripslashes($\_POST['Start']);

$endIndex = stripslashes($\_POST['End']);

if (isset($distances[$startIndex][$endIndex]))

{

echo "<p>The distance from " . $startIndex . " to " .

$endIndex . " is " . $distances[$startIndex][$endIndex]

. " kilometers, or " . round(($kmToMiles \* $distances[$startIndex][$endIndex]), 2) . " miles.</p>\n";

} else

{

echo "<p>The distance from " . $startIndex . " to " .

$endIndex . " is not in the array.</p>\n";

}

}

1. Add the following XHTML code after the PHP script to display the Web form:

<form action="[index].php" method="post">

<p>Starting City:

<select name="Start">

</select></p>

<p>Ending City:

<select name="End">

</select></p>

<p><input type="submit" name="submit"

value="Calculate Distance"/></p>

</form>

1. Add the following PHP script between the opening and closing XHTML tags for the ‘Start’ <select> element to insert the list of city names:

<?php

foreach ($distances as $city => $otherCities)

{

echo "<option value='$city'";

if (strcmp($startIndex,$city)==0)

{

echo " selected";  
}

echo ">$city</option>\n";

}

?>

1. Add the following PHP script between the opening and closing XHTML tags for the ‘End’ <select> element to insert the list of city names:

<?php

foreach ($distances as $city => $otherCities)

{

echo "<option value='$city'";

if (strcmp($endIndex,$city)==0)

{

echo " selected";

}

echo ">$city</option>\n";

}

?>

1. Save the document and open the file in your Web browser.
2. Make sure you understand the complete code and check whether the output is correct. If so, close your Web browser window, otherwise rewrite your code till it is correct.

### B1.2 Activiteit 2

For this assignment, you will create a multidimensional array that contains

the measurements, in inches, for several boxes that a shipping company

might use to determine the volume of a box.

1. Create a new document in your text editor.
2. Type the <!DOCTYPE> declaration, <html> element, document <head>, and <body> element and create a script section in the document body:

<?php

?>

1. Declare and initialize an associative multidimensional array using the information shown in the following table:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Length | Width | Depth |
| **Small Box** | 12 | 10 | 2.5 |
| **Medium Box** | 30 | 20 | 4 |
| **Large Box** | 60 | 40 | 11.5 |

1. Add statements to the end of the script section that display the volume

(length \* width \* depth) of each box in an efficient way (use a loop).

For example: *“The volume of the* ***Small Box*** *is:* ***300****.”.*

1. Save the document and open the file in your Web browser.
2. Check if the output is correct and if so, close your Web browser window, otherwise rewrite your code till it is correct.

### B1.3 Activiteit 3

Create a Song Organizer script that stores songs in a text file. Include functionality that allows users to view the song list and prevents the same song name from being entered twice. Also, include code that sorts the songs by name and genre, deletes duplicate entries, and randomizes the song list with the shuffle() function.

1. Create a new document in your text editor.
2. Type the <!DOCTYPE> declaration, <html> element, header information, and <body> element. Use “Song Organizer” as the content of the <title> element.
3. Add the following XHTML code and script section to the document body:

<h1>Song Organizer</h1>

<?php

?>

1. Add the following code to the script section to handle any parameters in the URL:

if (isset($\_GET['action']))

{

if ((file\_exists("SongOrganizer/songs.txt")) &&

(filesize("SongOrganizer/songs.txt")!= 0))

{

$SongArray = file("SongOrganizer/songs.txt");

switch ($\_GET['action']) {

} // End of the switch statement

}

}

1. Add the following code to the body of the switch statement to handle the three options (Sort Ascending, Remove Duplicates, and Shuffle):

case 'Remove Duplicates':

$SongArray = array\_unique($SongArray);

$SongArray = array\_values($SongArray);

break;

case 'Sort Ascending':

sort($SongArray);

break;

case 'Shuffle':

shuffle($SongArray);

break;

1. Add the following code immediately after the switch statement to save the song list after it has been modified:

if (count($SongArray)>0)

{

$NewSongs = implode($SongArray);

$SongStore = fopen("SongOrganizer/songs.txt","wb");

if ($SongStore === false)

{

echo "There was an error updating the song file\n";

} else {

fwrite($SongStore, $NewSongs);

fclose($SongStore);

}

} else {

unlink("SongOrganizer/songs.txt");

}

1. Add the following code to the end of the script section to handle any data submitted from the Web form:

if (isset($\_POST['submit']))

{

$SongToAdd = stripslashes($\_POST['SongName']) . "\n";

$ExistingSongs = array();

if (file\_exists("SongOrganizer/songs.txt") &&

filesize("SongOrganizer/songs.txt") > 0)

{

$ExistingSongs = file("SongOrganizer/songs.txt");

}

}

1. Add the following if statement immediately after the block where the song file data was read into the $ExistingSongs array. This if statement checks to see if the song name entered is already in the song list, and displays a message if the song already exists.

if (in\_array($SongToAdd, $ExistingSongs))

{

echo "<p>The song you entered already exists!<br>\n";

echo "Your song was not added to the list.</p>";

}

1. Add the following else clause to the preceding if statement. This else clause adds the new song to the song list file.

else {

$SongFile = fopen("SongOrganizer/songs.txt", "ab");

if ($SongFile === false)

{

echo "There was an error saving your message!\n";

} else {

fwrite($SongFile, $SongToAdd);

fclose($SongFile);

echo "Your song has been added to the list.\n";

}

}

1. Add the following code to the end of the script section to display the song list, or a message that there are no songs in the list if the list is empty:

if ((!file\_exists("SongOrganizer/songs.txt")) ||

(filesize("SongOrganizer/songs.txt") == 0))

{

echo "<p>There are no songs in the list.</p>\n";

} else {

$SongArray = file("SongOrganizer/songs.txt");

echo "<table border=\"1\" width=\"50%\"

style=\"background-color:lightgray\">\n";

echo "<th>Song</th> ";

foreach ($SongArray as $Song)

{

echo "<tr>\n";

echo "<td>" . htmlentities($Song) ."</td>";

echo "</tr>\n";

}

echo "</table>\n";

}

1. Add the following XHTML code immediately after the PHP script section to display hyperlinks for the three functions in the switch statement

(Sort Ascending, Remove Duplicates, and Shuffle):

<p>

<a href="[index].php?action=Sort%20Ascending">

Sort Song List</a><br>

<a href="[index].php?action=Remove%20Duplicates">

Remove Duplicate Songs</a><br>

<a href="[index].php?action=Shuffle">

Randomize Song list</a><br>

</p>

1. Next, add the following XHTML code to create a Web form for entering new song names into the song list:

<form action="[index].php" method="post">

<p><b>Add a New Song</b></p>

<p>Song Name: <input type="text" name="SongName"/></p>

<p>

<input type="submit" name="submit"

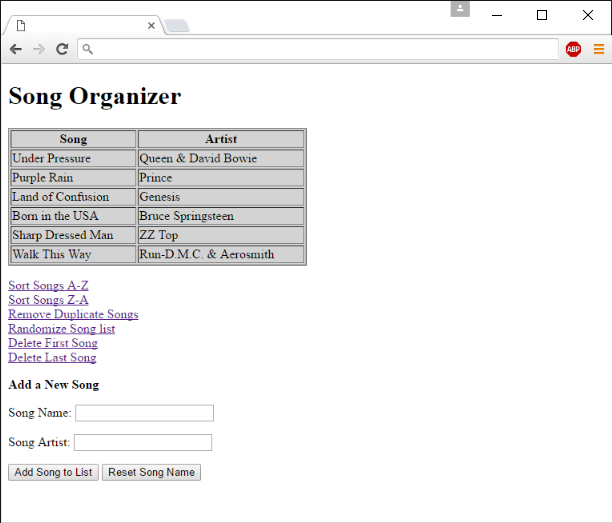
value="Add Song to List" />

<input type="reset" name="reset"

value="Reset Song Name" />

</p>

</form>

1. Create a ‘SongOrganizer’ folder in your current project, which should be on the same level as your php file.
2. Save the document and open the file in your Web browser. Make sure the output is correct, by testing all the features.
3. After you have made sure that the output is correct, modify the code by writing an if statement that prevents a user from adding a song when the input field has not been filled in.
4. Add a new input field for ‘artist’, of which the info is stored alongside the song name in a text file separated by a hyphen (-), and shown in the table. Make sure to modify the previous assignment, as this new input field must be filled in at all times too.
5. Write another action, one that sorts the songs descending.
6. **Write two more actions, one for deleting the first song and one for deleting the last song of the list. You should now have a web page that handles the actions and reading/writing correctly and looks similar to the following screenshot:

### B1.4 Activiteit 4

1. Create an All-in-One form that prompts the user to enter the 12 zodiac signs in random order in a text box. The signs must be separated with commas.
2. Inform the user that you will display the signs in an alphabetized list after the form has been submitted.
3. Use the array sort functions to alphabetize the signs and then display them in an ordered list.

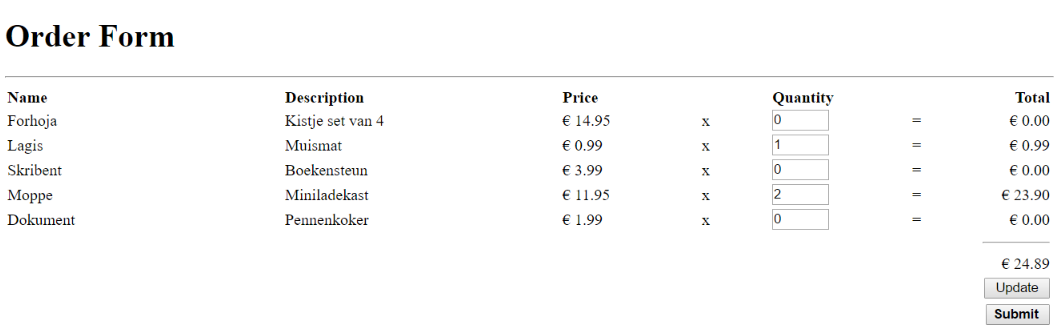
### B1.5 Activiteit 5

1. Search for images of zodiac signs online; 5 being the minimum.
2. Create a PHP script with an associative array that uses image filenames as the keys and image captions as the values.
3. Use the keys and values from the array to create a thumbnail gallery of images in which you display a small hyperlinked image to open a larger version of the image.
4. Set a width and height for the thumbnails that are less than half of the height and width of the images. Format the display of the thumbnail gallery.

### B1.6 Activiteit 6

Create an online order form as a Web form. Allow visitors to enter a quantity for at least five items for sale. Each item should have a name, description, and price. The form should have two buttons, one to update the totals for the quantities entered and one to submit the order. Save the orders in a ‘.txt’ file and store these in a subdirectory called ‘OnlineOrders’ in your Project directory. Use the date and time to create a unique filename for each order (each text file).

An example online order form:



*Tip: Make sure the quantities entered and the totals based on the entered info stay in place after pressing the update button, so when pressing the ‘Submit’ button the ‘.txt’ file does not come up empty.*