CS 328 - Homework 8

Deadline

11:59 pm on **Sunday, March 31, 2024**

Purpose

To get more practice with PHP, and with PHP documents that both create a form and respond to that form.

How to submit

Each time you wish to submit, submit your files using `~st10/328submit` on nrs-projects, with a homework number of `8`.

Important notes

- **NOTE**: you are welcome to use `require_once/include_once/require/include` in your PHP documents!
  - BUT when you use them in homework problems' documents, be sure to also SUBMIT copies of all files that you are requiring/including!
- Remember: You are required to use **`normalize.css`** for all of your web pages for CS 328, and to add **`link`** elements for additional CSS external stylesheets *after* this (but still **within** the `head` element).

  EXCEPT for `normalize.css`, **DO NOT USE ANY CSS FRAMEWORKS or PREDEFINED LIBRARIES for this course** (unless you get prior, explicit permission). One of this course's purposes is to provide you with some practice with the basics of "plain" CSS, so you can better make use of frameworks and predefined libraries later.
- Remember: There are now **CS 328 PHP Coding Standards so far** posted on the public course web site, under References -- you are also expected to follow these for all course PHP documents.
- You are expected to follow all course coding standards posted on the public course web site; course documents are also expected to validate as "strict"-style HTML, and valid CSS.

Problem 1

The purpose of this problem is to give you practice writing a PHP document with a loop, and more practice creating a PHP document that **both** creates a form **and** crafts a response to that form when it is submitted (as demonstrated in the posted Week 9 Lecture 2 example `php-form-handling.php`, and as you did in the Week 9 Lab Exercise's `328lab09.php`).

Note that I'm calling an `input` element with `type="number"` a number-field, and an `input` element with `type="submit"` a submit button.

Using the posted `html-template.html` as the initial basis, create a PHP document **`328hw8-1.php`** that meets the following requirements:

- Include your name and last modified date in its opening comment, **AND** the URL this can be run from.
  - (You *will* lose some credit if this URL does not work when I or the grader paste it into a browser!)
- Include an appropriate **`title`** element.
- Include an appropriate **`h1`** element.
- Include **your last name** within a **`p`** element that you add to the **`footer`** element.
- Its initial **`form`** element should have an **`action`** attribute whose action is:

  `"<?= htmlentities($_SERVER["PHP_SELF"], ENT_QUOTES) ?>"`

  ...and should contain at least:
  - one number-field,
  - at least ONE OTHER form control element of your choice
  - a submit button
- Its response should VISIBLY do something, based on the "other" form control element(s), the number of times entered into the number-field.
  - (This could be as simple as, for example, printing something entered into a textfield the number of times entered into the number-field, or the response could be something more involved -- you get to decide!)
  - The response should **ALSO** include an `a`/anchor element (hypertext link) with appropriate text that links back to `328hw8-1.php`.
- Make sure you appropriately **sanitize** ALL of the inputs provided by the user! (And remember that this care is necessary **regardless** of the form control in use -- the actual request might NOT be coming from your form!)
- It is up to you whether you would like to add additional CSS styling to this using an external CSS -- BUT if you do, also submit the `.css` file used.

Submit your file **`328hw8-1.php`** (and all additional files it uses, if any).

FUN FACT: HTML checkbox elements and PHP, Part 1

What you should already know about checkbox elements:

When a checkbox form widget such as the one in this fragment:

```
<input type="checkbox" name="game_desired" id="game_yes" />
<label for="game_yes">Check if you want to play a game</label>
```

...is within a form and that form is submitted, then:

- If that checkbox **IS** checked when the form is submitted, it sends the name=value pair:

  `game_desired=on`

...for that checkbox.

  - And, if that form had `method="post"`, then in a PHP is handling that submission, `$_POST["game_desired"]` would have the value `"on"`.

- If that checkbox is **NOT** checked when the form is submitted, it sends **NO** name=value pair for that checkbox.

Problem 2

The purpose of this problem is to give you a chance to try out PHP's **`array_key_exists`** function.

You can see **IF** an array **KEY** exists in a PHP associative array using:

`array_key_exists($desired_key, $desired_array)`

For example, for:

`$my_first_assoc_array = ["a" => 1, "b" => 2, "c" => 3];`

these would be true:

- `array_key_exists("a", $my_first_assoc_array) === true`
- `array_key_exists("j", $my_first_assoc_array) === false`

For this problem, you are going to create yet another PHP document that **both** creates a form **and** crafts a response to that form when it is submitted (as demonstrated in the posted Week 9 Lecture 2 example `php-form-handling.php`, and as you did in the Week 9 Lab Exercise's `328lab09.php`).

Using the posted `html-template.html` as the initial basis, create a PHP document **`328hw8-2.php`** that meets the following requirements:

- Include your name and last modified date in its opening comment, **AND** the URL this can be run from.
  - (You *will* lose some credit if this URL does not work when I or the grader paste it into a browser!)
- Include an appropriate **`title`** element.
- Include an appropriate **`h1`** element.
- Include **your last name** within a **`p`** element that you add to the **`footer`** element.
- Its initial **`form`** element should have an **`action`** attribute whose action is:

  `"<?= htmlentities($_SERVER["PHP_SELF"], ENT_QUOTES) ?>"`

  ...and should contain at least:

  - a **fieldset** with **at least four checkboxes**,

    each of which has a *DIFFERENT* **`name`** attribute.

    - You get to choose the "theme" for these checkboxes -- you need to use one that is different from any of the posted class examples.
  - a submit button

- In its response:
  - If at least one of the checkboxes was checked,
    - it should display a `ul` element with an `li` element for each selected checkbox.

    and if none of the checkboxes was checked,
    - it should display a `p` element whose content is a statement that none were selected.
  - You are expected to appropriately use the function **`array_key_exists`** in this part of your PHP document.
  - The response should **ALSO** include an `a`/anchor element (hypertext link) with appropriate text that links back to `328hw8-2.php`.
- Make sure you appropriately **sanitize** ALL of the inputs provided by the user! (And remember that this care is necessary **regardless** of the form control in use -- the actual request might NOT be coming from your form!)
- It is up to you whether you would like to add additional CSS styling to this using an external CSS -- BUT if you do, also submit the `.css` file used.

Submit your file **`328hw8-2.php`** (and all additional files it uses, if any).

FUN FACT: HTML checkbox elements and PHP, Part 2

Demonstrated in zyBooks Chapter 5 - Section 5.11 - Participation Activity 5.11.7:

IF:

- you have several "logically grouped" checkboxes
- you give them ALL the SAME **`name`** attribute that ends with **`[]`**
- you give them EACH a *different* **`value`** attribute with a value representing that checkbox's meaning

THEN:

- If at least ONE of those checkboxes is checked when the form is submitted,

  it sends a name=value pair where:
  - the name is that common `name` attribute's value without the `[]`,
  - and the value is an *array* containing the `value` attribute values for EACH checkbox that was checked at the time of submission!
- If NONE are checked, *no* name=value pair is sent.

That is, consider this fragment:

```
<input type="checkbox" name="transport[]" value="bicycle"
       id="bike" />
<label for="bike"> Bicycle </label> <br />
```

```
<input type="checkbox" name="transport[]" value="motorcar"
       id="car" />
<label for="car"> Car </label> <br />

<input type="checkbox" name="transport[]" value="skatebd"
       id="board" />
<label for="board"> Skateboard </label>
```

If this fragment is within a form with `method="post"` and that form is submitted, then:

- If, for example, the checkboxes for Bicycle and Skateboard are checked when the form is submitted, then `$_POST["transport"]` will contain an array whose values are `["bicycle", "skatebd"]`
- If NONE of these checkboxes with `name="transport[]"` checkbox is checked when the form is submitted, it sends NO name=value pair for any of those checkboxes.

Well, this needs to be tried out!

Problem 3

The purpose of this problem is to try out "logically grouped" checkboxes and how PHP can respond to a form containing them.

For this problem, you are going to create yet another PHP document that **both** creates a form **and** crafts a response to that form when it is submitted (as demonstrated in the posted Week 9 Lecture 2 example `php-form-handling.php`, and as you did in the Week 9 Lab Exercise's `328lab09.php`).

Using the posted `html-template.html` as the initial basis, create a PHP document **`328hw8-3.php`** that meets the following requirements:

- Include your name and last modified date in its opening comment, **AND** the URL this can be run from.
  - (You *will* lose some credit if this URL does not work when I or the grader paste it into a browser!)
- Include an appropriate **`title`** element.
- Include an appropriate **`h1`** element.
- Include **your last name** within a **`p`** element that you add to the **`footer`** element.
- Its initial **`form`** element should have an **`action`** attribute whose action is:

  `"<?= htmlentities($_SERVER["PHP_SELF"], ENT_QUOTES) ?>"`

  ...and should contain at least:
  - a **fieldset** with **at least four checkboxes**,

each of which has the *SAME* **name** attribute whose name ends with **[]**,

and each of which has a DIFFERENT **value** attribute whose value logically describes that choice.

- It is fine if these have the same "theme" as those in Problem 2, but make sure you implement them as described above, so that you practice with this kind of "logically grouped" checkboxes.

- a submit button

- In its response:
  - If at least one of the checkboxes was checked,
    - it should display a ul element with an li element for each selected checkbox.

    and if none of the checkboxes was checked,

    - it should display a p element whose content is a statement that none were selected.
  - You are expected to appropriately use the function **array_key_exists** in this part of your PHP document.
  - The response should **ALSO** include an a/anchor element (hypertext link) with appropriate text that links back to 328hw8-3.php.
- Make sure you appropriately **sanitize** ALL of the inputs provided by the user! (And remember that this care is necessary **regardless** of the form control in use -- the actual request might NOT be coming from your form!)
- It is up to you whether you would like to add additional CSS styling to this using an external CSS -- BUT if you do, also submit the .css file used.

Submit your file **328hw8-3.php** (and all additional files it uses, if any).