CSCI 4000 Assignment 2

Total estimated time for this assignment: **12 hours** (if you are a good programmer)

The main programming language for this assignment should be **PHP** (mixed with HTML and CSS). If you do not use **PHP**, you will get **0 points**.

When you see "Richard Ricardo" in the example screen captures, change it to **<your name>**.
When you see "Richard" in the example screen captures, change it to **<your first name>**.
If you do not put **<your name> / <your first name>** in the above mentioned fields, you will get **0 points** for the question(s).

No two students should submit webpages with exactly the same code, content, layout, or color combination. If found, **both** students will get **0 points**.

Create a folder on your hard disk, name the folder **lastname_firstname_assignment2.** Save all the files from this assignment in this folder.

Create the following subfolders (in the folder **lastname_firstname_assignment2): q1, q2, q3, q4**.
As a result, you should have the following folder (directory) structure for this assignment:

- lastname_firstname_assignment2\q1\
- lastname_firstname_assignment2\q2\
- lastname_firstname_assignment2\q3\
- lastname_firstname_assignment2\q4\

Use XAMPP web server solution stack package to help debugging PHP code. It will make your debugging process easier. All php files must not produce any error, or any warning **(-2 points for each error, each warning**). Your program must run. A program that does not run will get at most **50% of the possible points**. All files must begin with a **comment** that identifies the author, the course code, and the program date **(- 2 points each question** if found missing). All html, css and php files must be clearly **documented (commented)**. Points will be taken off **(-2 points each question)** for insufficient comments (`<!-- -->, /* */, //`).

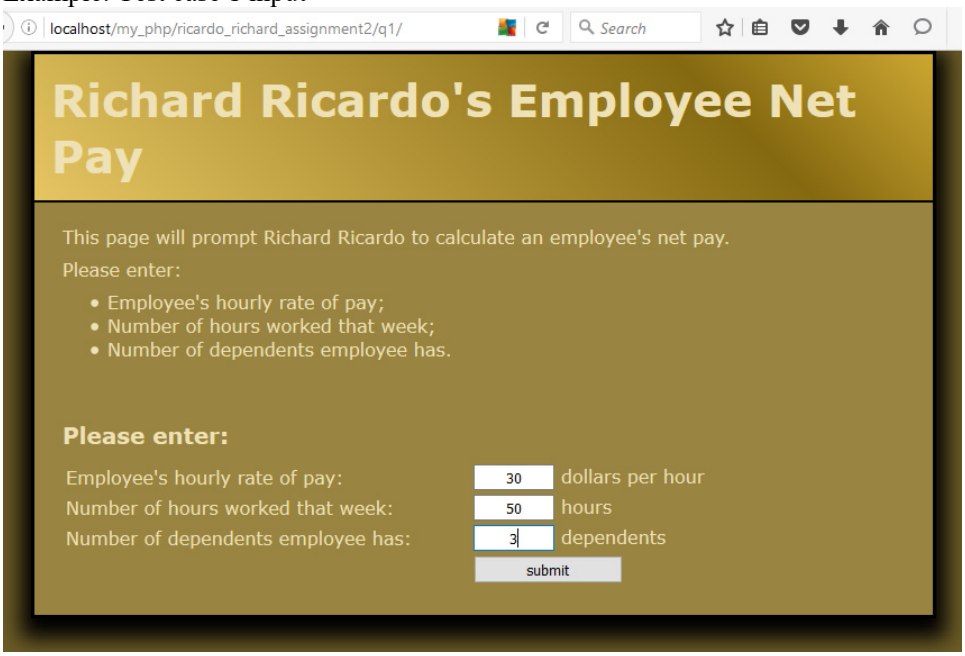
- Before adding PHP code, all html files must pass html validation at http://validator.w3.org/ without any **error** (and with only 1 warning).
- When you view page source in a web browser, **`<!DOCTYPE html>`** must be at the top of every page. In other words, all pages must be written in HTML5. **(-20 points** if not)
  - You **can** put php code before **`<!DOCTYPE html>`**.
  - You **cannot** put html code before **`<!DOCTYPE html>`**.
- All css files must pass css validation at http://jigsaw.w3.org/css-validator/ without any **error**. **(-2 points for each error/warning**, only 1 warning is allowed for html validator)

Question 1 – PHP Chapter 2 & 8 (25 points) **Estimated time: 3 hours**

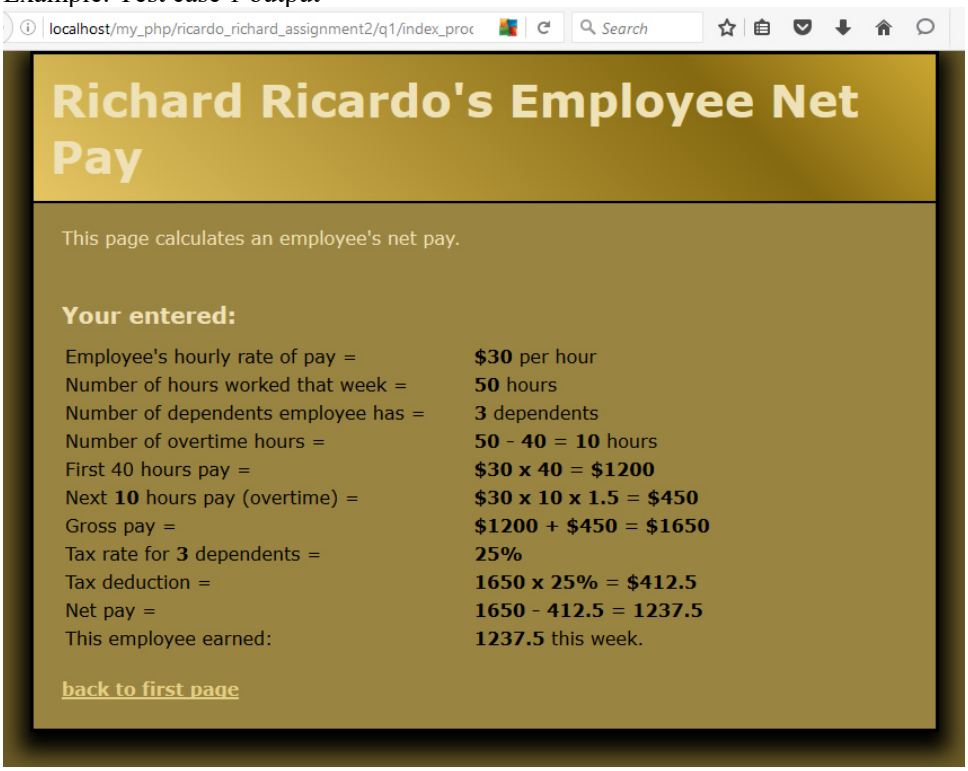
- Save the files in subfolder "**lastname_firstname_assignment2\q1**". (1 point)
- Create web page(s) that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below. (3 points)
- Create your page(s) using "**<your name>**'s Employee Net Pay**"** as the page title(s) (<title> tag). (2 points)
- Save the first page as **index.htm**. Use a HTML form to capture user input. Submit the form to a second file **index_process.php** using the **GET method**. (3 points)
- The second page **index_process.php** will then be **updated** as shown. It contains a link to go back to the first page. (2 points for link)
- Write a program that calculated an employee's net pay. (13 points for logic and display)
    - The user should enter
        - Hourly pay rate
        - Number of hours worked in a week
        - Number of dependents claimed
    - If the employee works more than 40 hours in a week, overtime is calculated at 1.5 times the regular hourly rate.
    - Taxes are then deducted from the gross pay as follows:
        - No dependents: tax rate is 28%
        - 1 to 3 dependents: tax rate is 25%
        - 4 to 6 dependents: tax rate is 15%
        - More than 6 dependents: tax rate is 10%
- Create a css file named **style.css** to format all pages by creating your own layout (no two students should have the same layout). You can use **the same (or similar) css file(s)** to format all questions. (3 points)

Input test case 1	**Output test case 1 and logic**
Employee's hourly rate: $30. Number of hours worked this week: 50 hours. Number of dependents: 3.	First 40 hours pay = $30 x 40 hours = $1200 Next 10 hours pay (overtime) = $30 x 10 hours x 1.5 = $450 Gross pay = $1200 + $450 = $1650 Tax rate for 3 dependents = 25% Tax deduction = $1650 x 25% = $412.5 Net pay = $1650 - $412.5 = $1237.5 This employee earned: $1237.5 this week.
Input test case 2	**Output test case 2 and logic**
Employee's hourly rate: $40. Number of hours worked this week: 30 hours. Number of dependents: 4.	First 40 hours pay = $40 x 30 hours = $1200 Next 10 hours pay (overtime) = $30 x 0 hours x 1.5 = $0 Gross pay = $1200 + $0 = $1200 Tax rate for 4 dependents = 15% Tax deduction = $1200 x 15% = $180 Net pay = $1200 - $180 = $1020 This employee earned: $1020 this week.

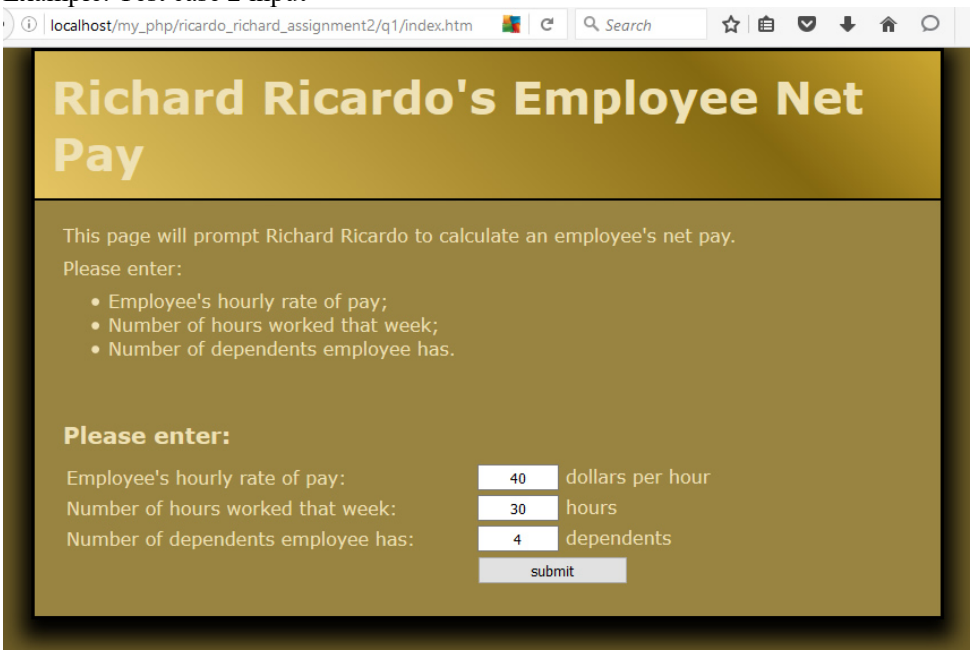
Example: Test case 1 input



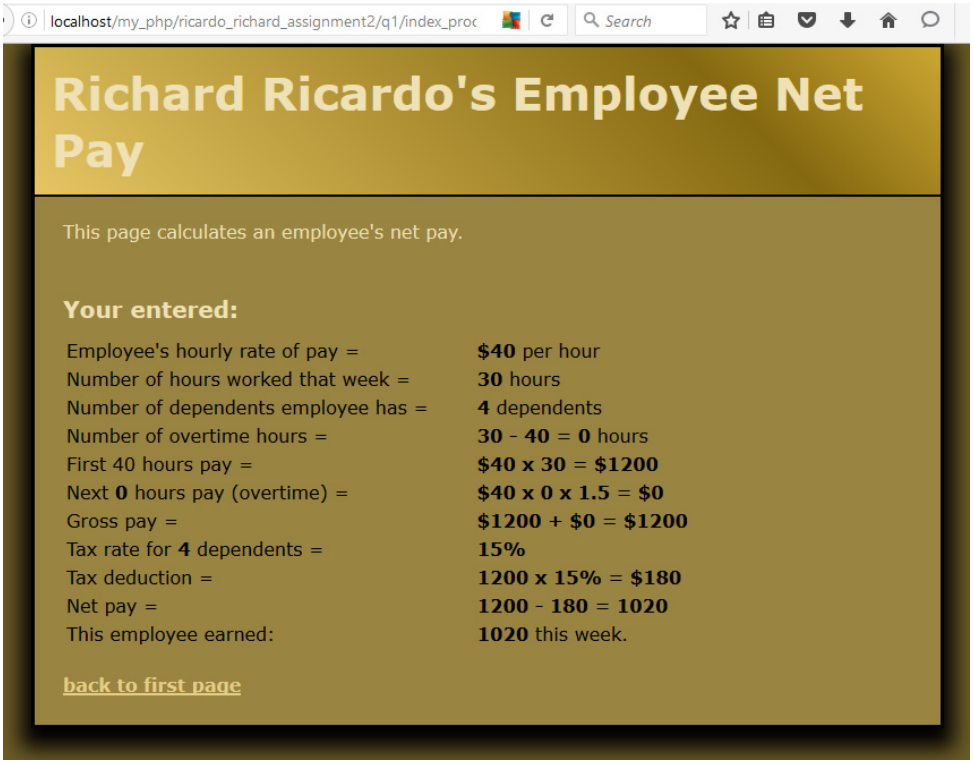
Example: Test case 1 output



Example: Test case 2 input



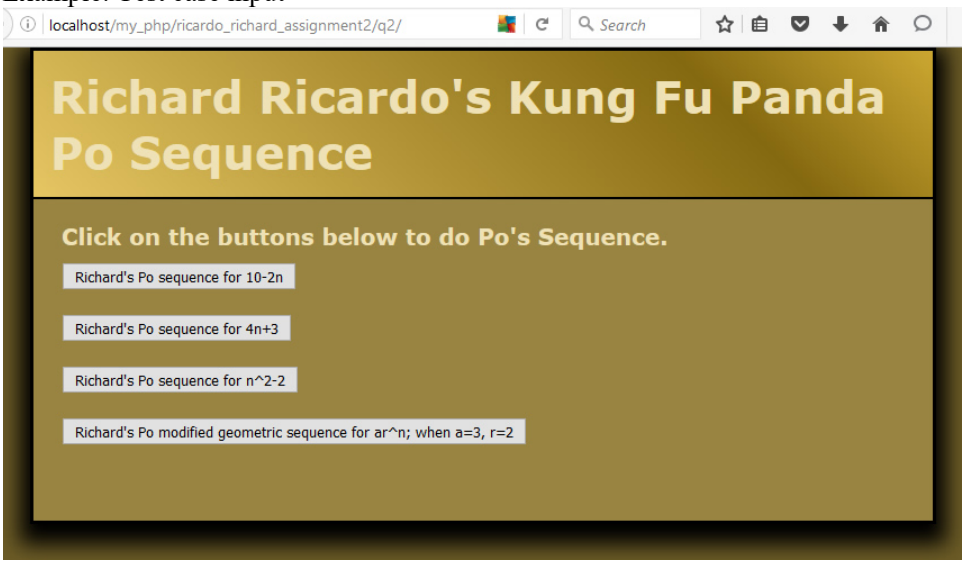
Example: Test case 2 output



Question 2 – PHP Chapter 2 & 8 (25 points) **Estimated time: 3 hours**

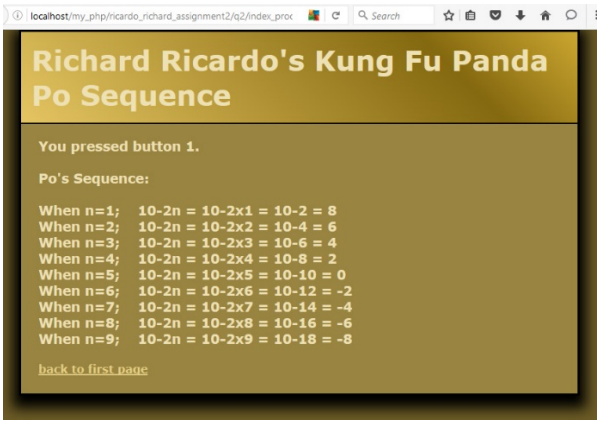
- Save the files in subfolder "**lastname_firstname_assignment2\q2**". (1 point)
- Create web page(s) that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below. (3 points)
- Create your page(s) using "**<your name>**'s Kung Fu Panda Po Sequence**"** as the page title(s) (<title> tag). (2 points)
- Save the first page as **index.htm (or index.php)**. Use a HTML form to capture user input. Submit the form to a second file **index_process.php** using the **POST method**. (3 points)
- The second page contains a link to go back to the first page. (2 points for link)
- Allow the user to click on one of the four buttons. (13 points)
    - You must use **for** loop, **while** loop or **do..while** loop for coding.
    - The sequences generated and the output should look like the example output shown.
- Create a css file named **style.css** to format all pages by creating your own layout (no two students should have the same layout). You can use **the same (or similar) css file(s)** to format all questions. (3 points)

Example: Test case input

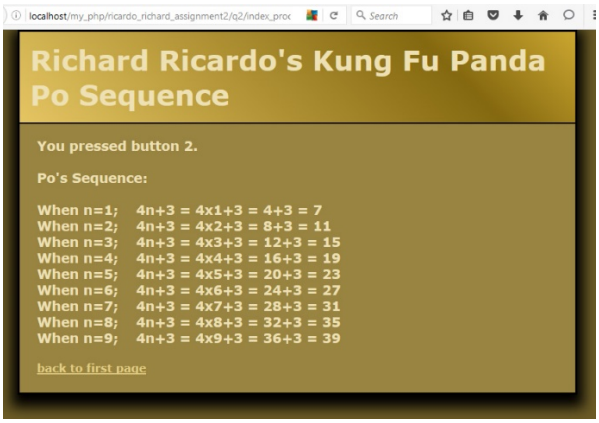


Example: Test case output

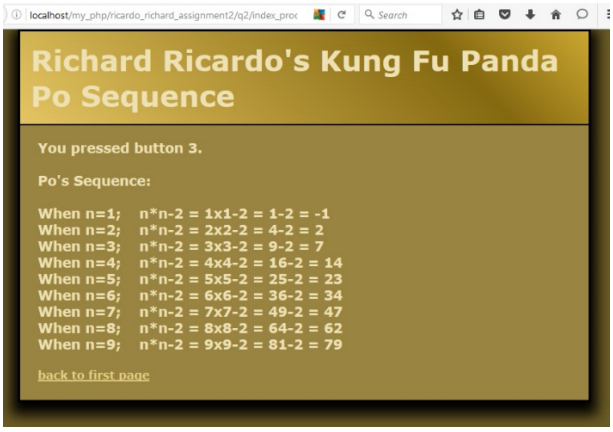
When button 1 is clicked



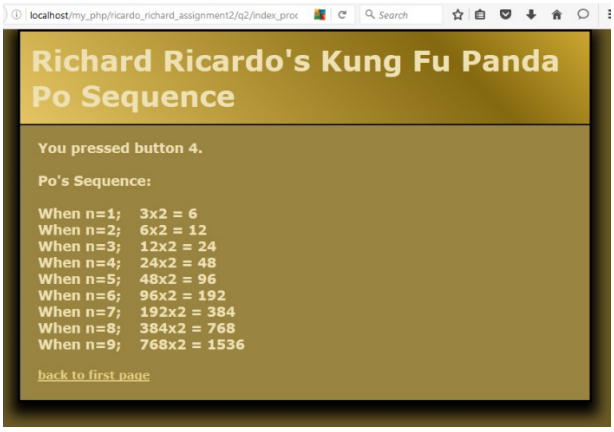
When button 2 is clicked



When button 3 is clicked



When button 4 is clicked



Question 3 – PHP Chapter 2 & 8 (25 points) **Estimated time: 3 hours**

- Save the files in subfolder "**lastname_firstname_assignment2\q3**". (1 point)
- Create web page(s) that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below. (3 points)
- Create your page(s) using "**<your name>**'s Kung Fu Panda Po Magic Rectangle**"** as the page title(s) (<title> tag). (2 points)
- Save the first page as **index.htm (or index.php)**. Use a HTML form to capture user input. Submit the form to a second file **index_process.php** using the **POST method**. (3 points)
- The second page contains a link to go back to the first page. (2 points for link)
- Write a program that asks the user for Po's row and column numbers for generating magic rectangles.
  - Capture row and column numbers. (1 point)
  - Generate the magic rectangles shown using **nested loops** (**two-dimensional loops**).
  - The first rectangle and related **calculation** must be generated by **nested for** loops. (4 points)

```
for (…)
{
        for (…) {}
}
```

  - The second rectangle must be generated by **nested while** loops. (4 points)

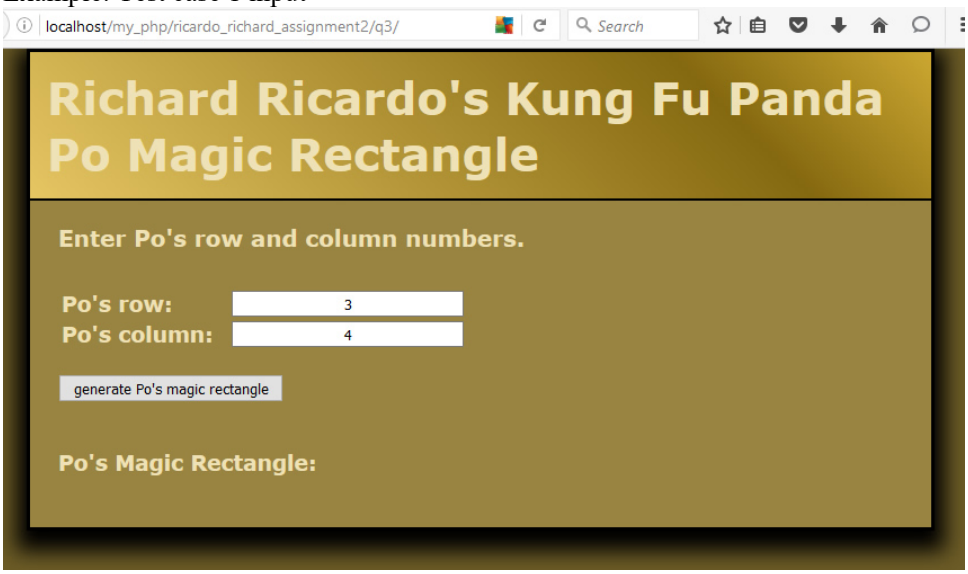
```
while (…)
{
        while (…) {}
}
```

  - The third rectangle must be generated by **nested do…while** loops. (4 points)

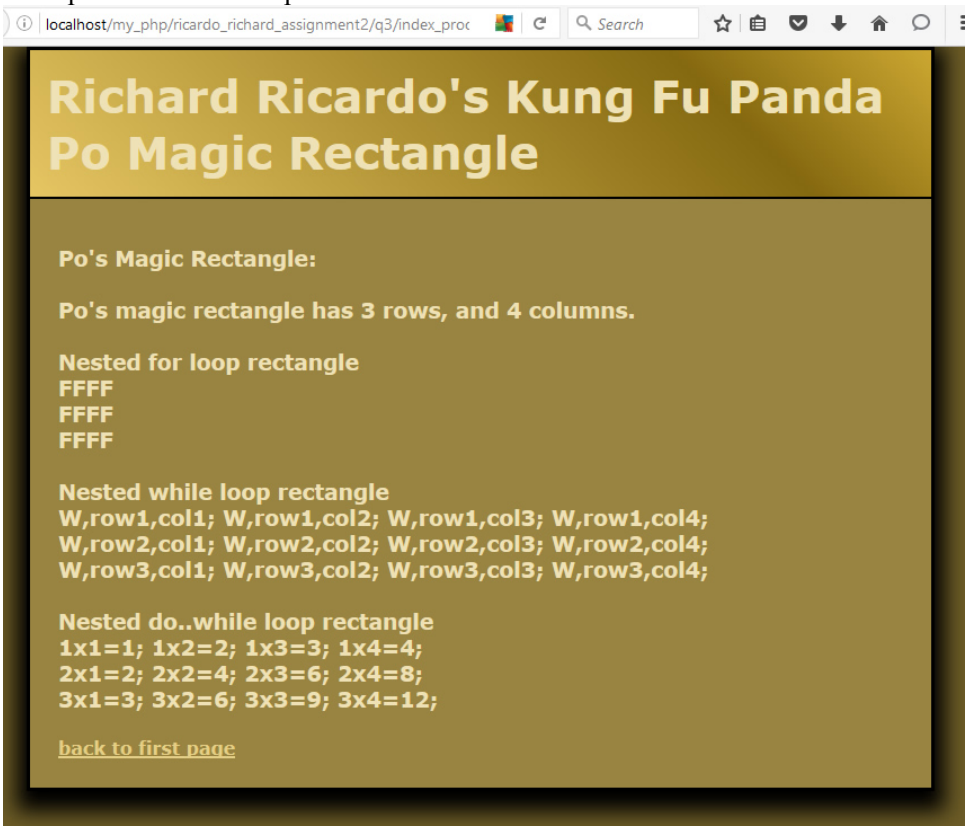
```
do
{
        do {} while (…)
}
```

- Create a css file named **style.css** to format all pages by creating your own layout (no two students should have the same layout). You can use **the same (or similar) css file(s)** to format all questions. (3 points)

Example: Test case 1 input

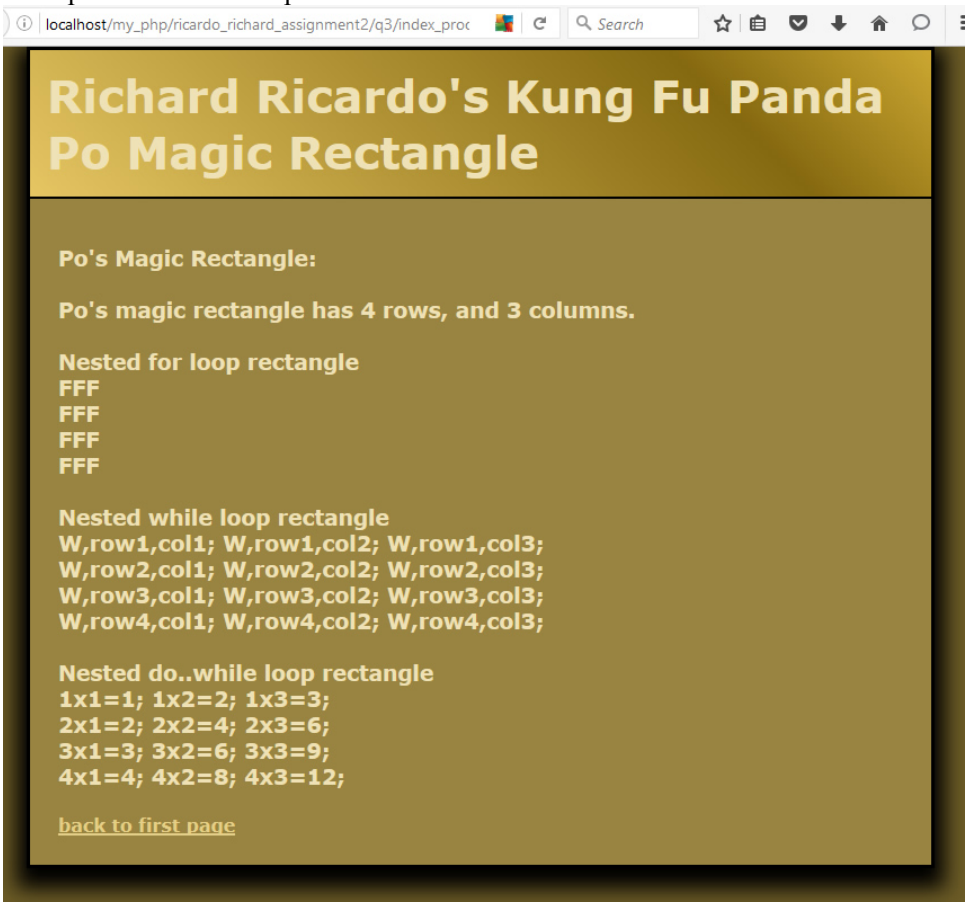


Example: Test case 1 output



Example: Test case 2 input

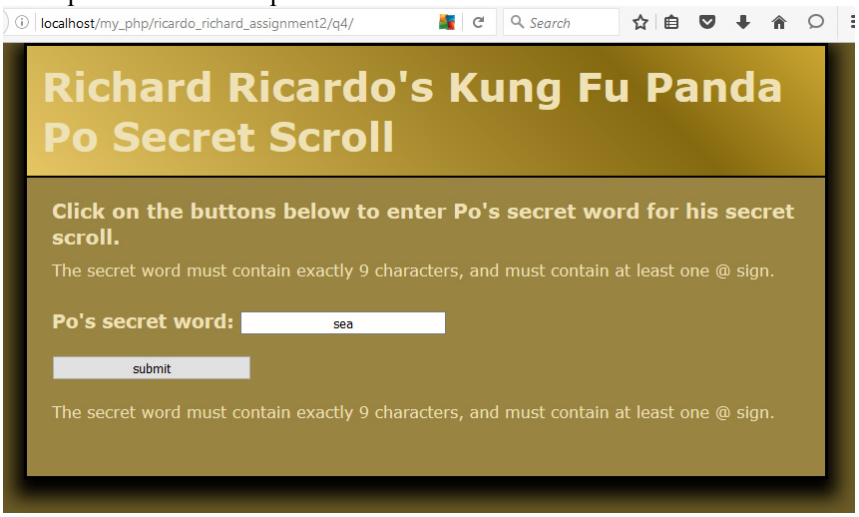
Example: Test case 2 output



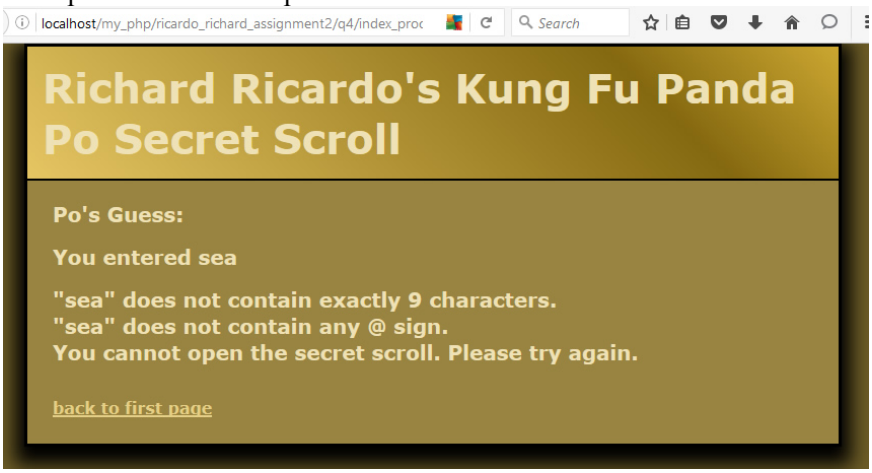
Question 4 – PHP Chapter 2 & 8 & 9 (25 points) **Estimated time: 3 hours**

- Save the files in subfolder "**lastname_firstname_assignment2\q4**". (1 point)
- Create web page(s) that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below. (3 points)
- Create your page(s) using "**<your name>**'s Kung Fu Panda Po Secret Scroll**"** as the page title(s) (<title> tag). (2 points)
- Save the first page as **index.htm (or index.php)**. Use a HTML form to capture user input. Submit the form to a second file **index_process.php** using the **GET method**. (3 points)
- The second page contains a link to go back to the first page. (2 points for link)
- Write a program that asks the user for Po's secret word. (13 points)
    - The secret word must have **exactly 9 characters** and **must contain at least one @ sign**. A bit of online research is needed here. Be creative in your problem solving approach.
    - All other keyboard characters are allowed.
    - You need to feedback to the user what is wrong with the secret word (error message as shown).
    - You do NOT need to ask user to re-enter secret word. You only need to show if the secret word is correct.
- Create a css file named **style.css** to format all pages by creating your own layout (no two students should have the same layout). You can use **the same (or similar) css file(s)** to format all questions. (3 points)

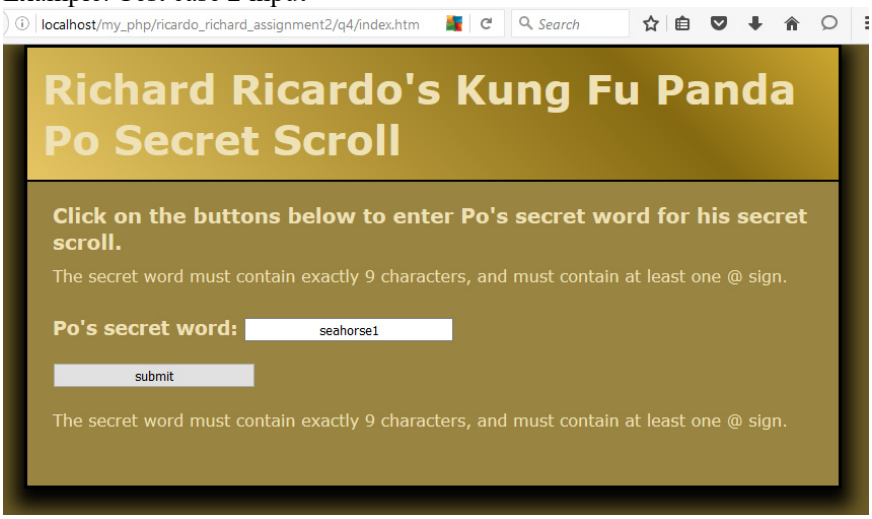
Example: Test case 1 input



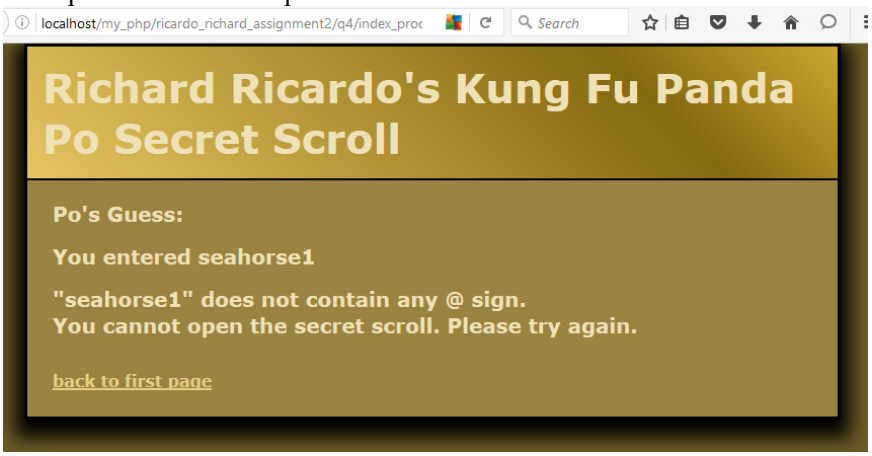
Example: Test case 1 output



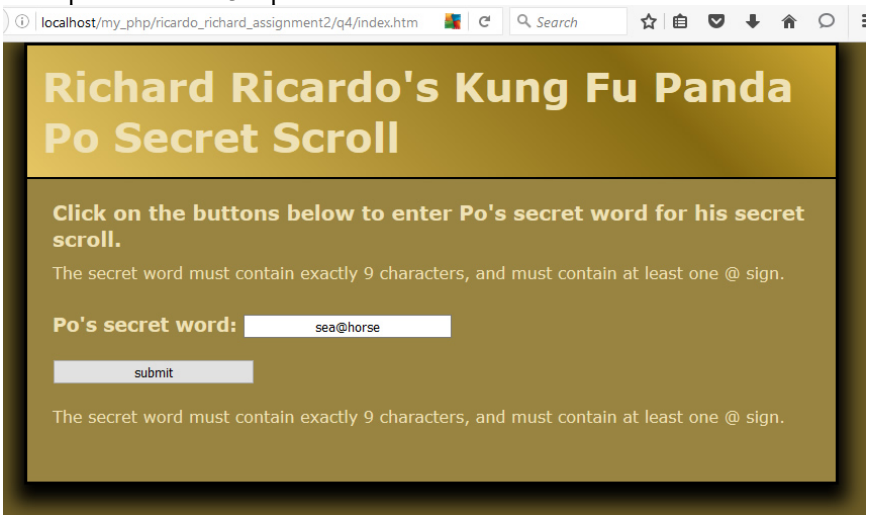
Example: Test case 2 input



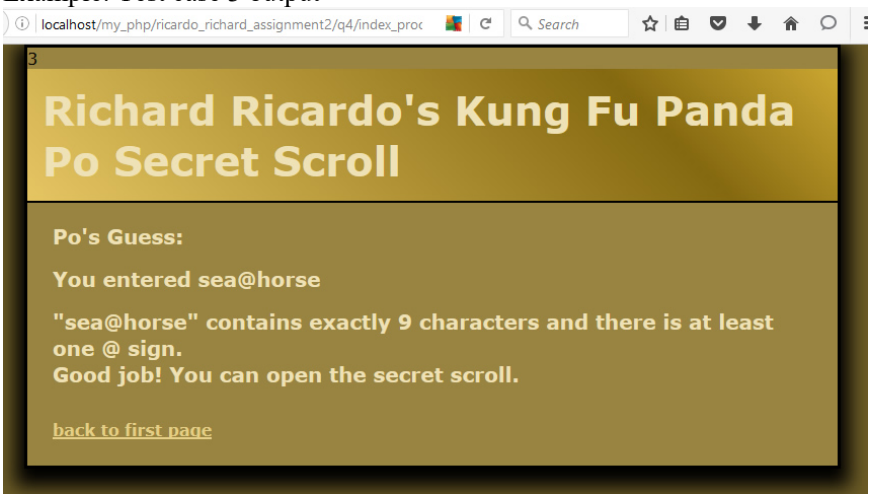
Example: Test case 2 output



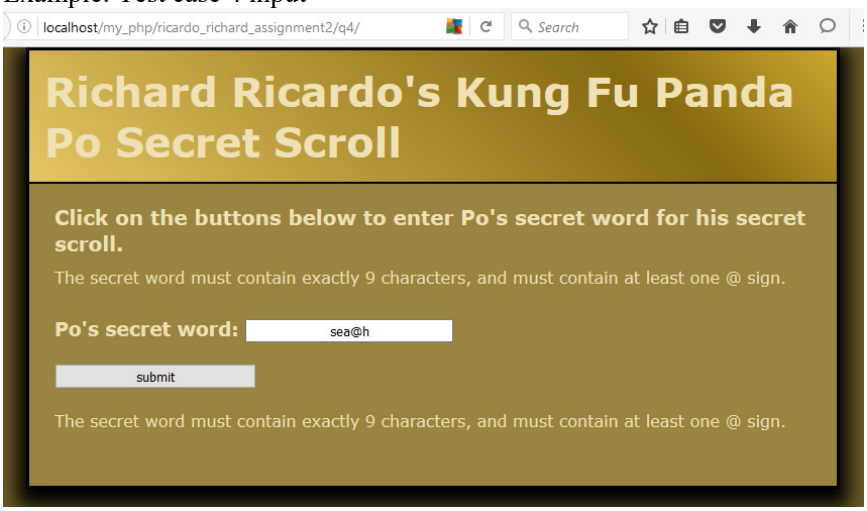
Example: Test case 3 input



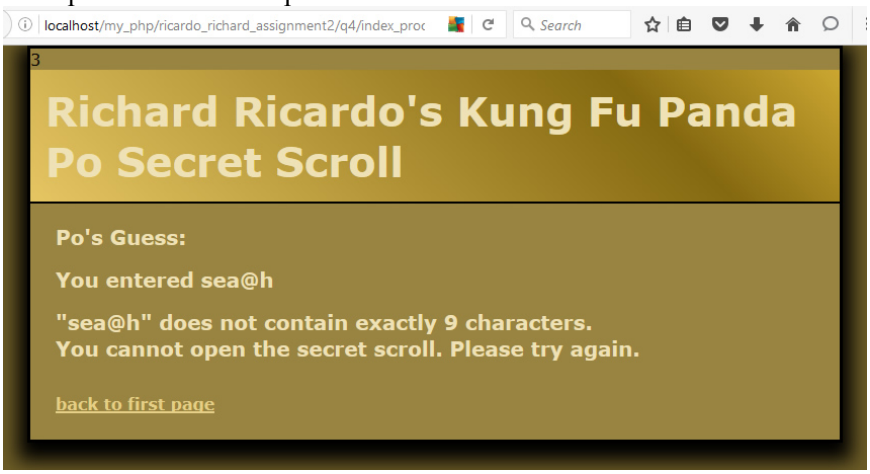
Example: Test case 3 output



Example: Test case 4 input



Example: Test case 4 output



Important:

1. If you do not put **<your name> / <your first name>** in the above mentioned fields (as shown in the examples), you will get **0 points** for the question(s).
2. **No two students** should submit webpages with exactly the same code, content, layout, or color combination. If found, both students will get **0 points**.
3. Before adding PHP code, all html files must pass html validation at http://validator.w3.org/ without any **error** (and with only 1 warning). Use the validator's "File Upload" tab to check each file.
    a. index.htm file must be validated.
    b. When you view page source in a web browser, **`<!DOCTYPE html>`** must be at the top of every page. In other words, all pages must be written in HTML5. **(-20 points** if not)
    c. You don't need to validate index_process.php after adding php code. If any html error is found, 2 points will be deducted for each error. Please validate the file before adding PHP code.
4. All css files must pass css validation at http://jigsaw.w3.org/css-validator/ without any **error**.
5. If your files do not pass the html and css validations, **2 points will be deducted** for **each html or css error** found (no deduction for php error at html validation).
6. Document (comment) your HTML files (`<!-- -->`), CSS files (`/* */`), and PHP files (`/* */` OR `//`). **Points will be taken off** for insufficient comments (`<!-- -->, /* */, //`).

Submission instructions:

- You need to test all document(s).
- Do screen capture(s) of the **input** and the related **output(s)**. Use any graphic editing software (e.g. Microsoft Paint, Adobe Fireworks, GIMP, or Microsoft Expression Design etc) to cut out the browser output (from the screen capture), paste them into a word document.
- Provide **2 different test cases** for each question. In other words, for **each question**, you may need to have **2 input** screen captures and **2 related output** screen captures.
- Do screen capture(s) of html validation results (index.htm files only) and css validation results (style.css file only), cut and paste them into the word document. If you use the same style.css file, you only need to do screen capture of one style.css file.
- Save the word document as a pdf file.

You need to submit the following:
1. A pdf file containing the screen capture(s) of the web browser input and output pages, and the screen capture(s) of all html validation results (from http://validator.w3.org/) and css validation results (from http://jigsaw.w3.org/css-validator/), name the file **lastname_firstname_assignment2.pdf**.
2. All html file(s), php file(s), css file(s), and other related files (e.g. image files). Zip your file folder (**lastname_firstname_assignment2**) into a single zip file (or rar file) **lastname_firstname_assignment2.zip**. In the above example, the zip file should contain the following files and subfolders. If there is any image, there should be a \images\ subfolder.

- lastname_firstname_assignment2\q1\index.htm
- lastname_firstname_assignment2\q1\index_process.php
- lastname_firstname_assignment2\q1\style.css // you may put style.css in a subfolder
- lastname_firstname_assignment2\q2\index.htm
- lastname_firstname_assignment2\q2\index_process.php
- lastname_firstname_assignment2\q2\style.css
- lastname_firstname_assignment2\q3\index.htm
- lastname_firstname_assignment2\q3\index_process.php
- lastname_firstname_assignment2\q3\style.css
- lastname_firstname_assignment2\q4\index.htm
- lastname_firstname_assignment2\q4\index_process.php
- lastname_firstname_assignment2\q4\style.css

Please submit the above mentioned **two files** (.pdf and .zip) to D2L digital dropbox.

Grading guidelines (programming questions):
Your programs will be judged on several criteria, which are shown below.

- Correctness (50%): Does the program compile (run) correctly? Does the program do what it's supposed to do?
- Design (20%): Are operations broken down in a reasonable way (e.g. classes and methods)?
- Style (10%): Is the program **indented** properly? Do variables have **meaningful names**?
- Robustness (10%): Does the program handle erroneous or unexpected input gracefully?
- Documentation (10%): Do all program files begin with a **comment** that identifies the author, the course code, and the program date? Are all the classes, methods and data fields clearly **documented (commented)?** Are unclear parts of code **documented (commented)?** (Some items mentioned may not apply to some languages)

A program that does not compile (run) will get at most **50% of the possible points**.