

## CSCI 4000 Assignment 1

**Total estimated time** for this assignment: **9 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**.

Please change the provided example’s layout and color combination (color scheme). **If you use (copy) ANY of the provide example’s color combination, 10 points will be deducted (-10 points).**

Create a folder on your hard disk, name the folder **lastname\_firstname\_assignment1**. Save all the files from this assignment in this folder.

Create the following subfolders (in the folder **lastname\_firstname\_assignment1**): **q1, q2, q3**.

As a result, you should have the following folder (directory) structure for this assignment:

- lastname\_firstname\_assignment1\q1\
- lastname\_firstname\_assignment1\q2\
- lastname\_firstname\_assignment1\q3\

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 (`<!-- -->`, `/* */`, `//`).

- 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>`.
- Before adding PHP code, all html files must pass html validation at <http://validator.w3.org/> without any **error** (and with only 1 warning).
- After adding PHP code, the generated html code (Firefox web browser > right-click > view page source) must also pass html validation at <http://validator.w3.org/> without any **error** (and with only 1 warning).
- 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 (33 points)****Estimated time: 3 hours**

- Save the files in subfolder “**lastname\_firstname\_assignment1\q1**”. (1 point)
- Create a web page to capture information about **your vehicle**.
- The initial page should look like the one shown below. (3 points)
- **Allow** the user to **enter the information** to the five textboxes. (5 points)
- Create your pages using “**<your name>**’s Vehicle Page” as the page titles (<title> tag). (2 points)
- Save the first page as **index.htm** (2 points). Use a HTML form to capture user input. Submit the form to a second file **index\_process.php** using the **POST Method**. (5 points)
- The second page **index\_process.php** will then be **updated** as shown. It contains a link to go back to the first page. (10 points)
- 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. (5 points)

Example: Test case 1 input

The screenshot shows a web browser window with the URL `localhost/my_php/ricardo_richard_assignment1/q1/`. The page has a dark olive green header with the title "Richard Ricardo's Vehicle". Below the header, the text reads: "This page will prompt Richard Ricardo for information on his dream vehicle. Click on the button below to enter new information". The form contains five input fields: "Vehicle Type:" with "Sedan", "Color:" with "Orange", "Year:" with "2012", "Make:" with "Ford", and "Model:" with "Focus". A "submit" button is located at the bottom left of the form area.

Example: Test case 1 output

The screenshot shows a web browser window with the URL `localhost/my_php/ricardo_richard_assignment1/q1/index_proc`. The page has the same dark olive green header and title "Richard Ricardo's Vehicle". The text reads: "This page displays the information of Richard Ricardo's dream vehicle." Below this, the information is displayed as a list of items: "• Vehicle Type: Sedan", "• Color: Orange", "• Year: 2012", "• Make: Ford", and "• Model: Focus". At the bottom left, there is a link labeled "back to first page".

## Example: Test case 2 input

The screenshot shows a web browser window with the address bar containing 'localhost/my\_php/ricardo\_richard\_assignment1/q1/'. The page title is 'Richard Ricardo's Vehicle'. The main content area has a dark olive green background with a yellow header bar containing the title. Below the header, there is a message: 'This page will prompt Richard Ricardo for information on his dream vehicle.' followed by the instruction 'Click on the button below to enter new information'. The form consists of five rows, each with a label on the left and a text input field on the right: 'Vehicle Type:' with 'Sports Car', 'Color:' with 'Red', 'Year:' with '2013', 'Make:' with 'Lamborghini', and 'Model:' with 'Aventador LP 720-4'. At the bottom of the form is a 'submit' button.

Richard Ricardo's Vehicle

This page will prompt Richard Ricardo for information on his dream vehicle.

Click on the button below to enter new information

Vehicle Type: Sports Car

Color: Red

Year: 2013

Make: Lamborghini

Model: Aventador LP 720-4

submit

## Example: Test case 2 output

The screenshot shows the same web browser window, but the page content has changed. The header and title remain the same. The main message now reads: 'This page displays the information of Richard Ricardo's dream vehicle.' Below this, the form data is displayed as a list of items, each with a bullet point: '• Vehicle Type: Sports Car', '• Color: Red', '• Year: 2013', '• Make: Lamborghini', and '• Model: Aventador LP 720-4'. At the bottom of the page, there is a link labeled 'back to first page'.

Richard Ricardo's Vehicle

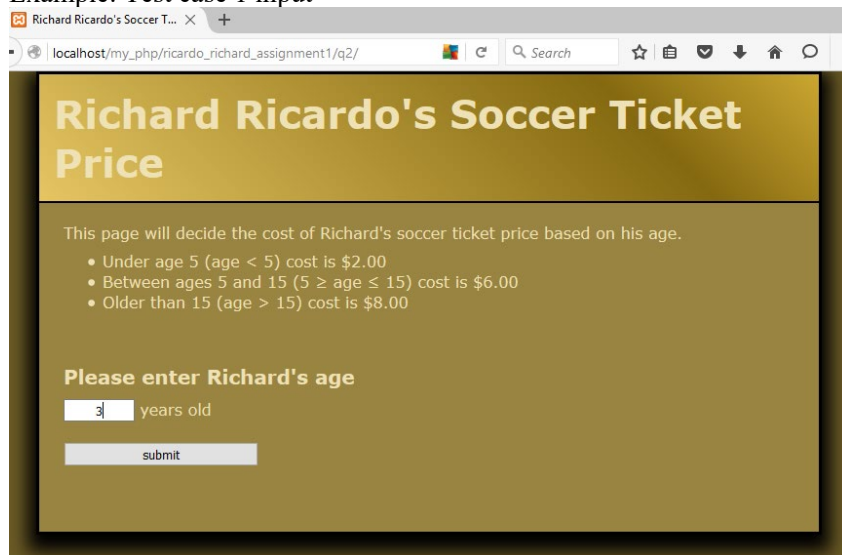
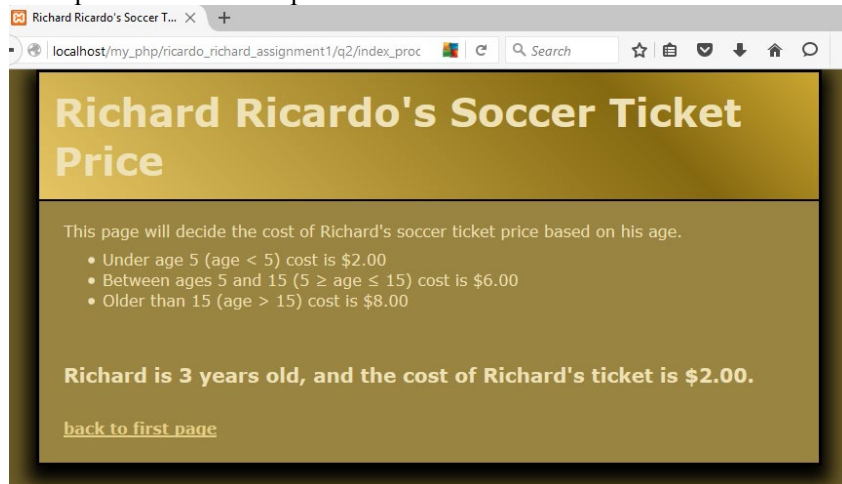
This page displays the information of Richard Ricardo's dream vehicle.

- Vehicle Type: Sports Car
- Color: Red
- Year: 2013
- Make: Lamborghini
- Model: Aventador LP 720-4

[back to first page](#)

**Question 2 – PHP Chapter 2 & 8 (33 points)****Estimated time: 3 hours**

- Save the files in subfolder “**lastname\_firstname\_assignment1\q2\**”. (1 point)
- Create web page(s) that displays the cost of a soccer ticket as shown. (4 points)
- Allow user to enter his or her **age**. (2 points)
- The output should be a message telling the user what the customer’s soccer ticket will cost, based on the following criteria: (12 points for logic)
  - Under age 5 (age < 5), cost is \$2
  - Between ages 5 and 15, cost is \$6
  - Older than 15 (age > 15), cost is \$8
- Create your pages using “<b>your name</b>’s Soccer Ticket Price” as the page titles (<title> tag). (4 points)
- Save the first page as **index.htm**. Use an HTML form to capture user input. Submit the form to a second file **index\_process.php** using the **GET method**. (5 points)
- The second page **index\_process.php** will then be **updated** as shown. It contains a link to go back to the first page. (5 points)
- 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. (5 points)

**Example: Test case 1 input****Example: Test case 1 output**

## Example: Test case 2 input

The screenshot shows a web browser window with the URL `localhost/my_php/ricardo_richard_assignment1/q2/index.htm`. The page title is "Richard Ricardo's Soccer Ticket Price". The content includes a heading, a paragraph explaining the page's purpose, a bulleted list of age-based pricing rules, a form prompt "Please enter Richard's age", an input field containing "10", and a "submit" button.

## Example: Test case 2 output

The screenshot shows the same web browser window after the form is submitted. The page now displays the calculated result: "Richard is 10 years old, and the cost of Richard's ticket is \$6.00." Below this, there is a link labeled "back to first page".

## Question 3 – PHP Chapter 2 &amp; 8 (34 points)

Estimated time: 3 hours

- Save the files in subfolder “**lastname\_firstname\_assignment1\q3**”. (1 point)
- Create a web page that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below. (4 points)
- Create your pages using “**<your name>**’s Kung Fu Panda Po Power Level” as the page titles (`<title>` tag). (4 points)
- Allow the user to **enter a number** representing Kung Fu Panda Po's power level (2 points).
- The power level conversions are as follows: (12 points for logic)
  - less than 0: Po eats too much noodle
  - 0 to 200: Po reaches the training hall
  - 201 to 400: Po reaches the student barracks
  - 401 to 600: Po reaches the Peach Tree of Heavenly Wisdom
  - 601 to 800: Po climbs the Wu Da Mountains
  - 801 to 1000: Po trains at the Pool of Sacred Tears
  - Greater than 1000: Po defeats Tai Lung

- 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**. (5 points)
- The second page **index\_process.php** will then be **updated** as shown. It contains a link to go back to the first page. (5 points)
- 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. (5 points)

### Example: Test case 1 input

The screenshot shows a web browser window with the address bar displaying 'localhost/my\_php/ricardo\_richard\_assignment1/q3/index.htm'. The page has a yellow header with the title 'Richard Ricardo's Kung Fu Panda Po Power Level'. Below the header, there is a paragraph of text: 'This page will prompt Richard Ricardo for Kung Fu Panda Po's power level (in number). The program will convert the given input into Po's power metric.' This is followed by a list of power level ranges and their corresponding achievements: 'less than 0: Po eats too much noodle', '0 to 200: Po reaches the training hall', '201 to 400: Po reaches the student barracks', '401 to 600: Po reaches the Peach Tree of Heavenly Wisdom', '601 to 800: Po climbs the Wu Da Mountains', '801 to 1000: Po trains at the Pool of Sacred Tears', and 'Greater than 1000: Po defeats Tai Lung'. Below this list, there is a prompt: 'Click on the button below to submit Po's power level.' Underneath, there is a label 'Po's power level:' followed by a text input field containing the number '888'. At the bottom, there is a 'submit' button.

### Example: Test case 1 output

The screenshot shows the same web browser window as the input page, but the page content has changed. The header and the list of power level ranges remain the same. However, the text below the list now reads: 'Po's power level is 12.' followed by 'Po reaches the training hall.' At the bottom of the page, there is a link labeled 'back to first page'.

### Example: Test case 2 input

Richard Ricardo's Kung Fu Panda Po Power Level

This page will prompt Richard Ricardo for Kung Fu Panda Po's power level (in number). The program will convert the given input into Po's power metric.

less than 0: Po eats too much noodle  
0 to 200: Po reaches the training hall  
201 to 400: Po reaches the student barracks  
401 to 600: Po reaches the Peach Tree of Heavenly Wisdom  
601 to 800: Po climbs the Wu Da Mountains  
801 to 1000: Po trains at the Pool of Sacred Tears  
Greater than 1000: Po defeats Tai Lung

**Click on the button below to submit Po's power level.**

Po's power level:

### Example: Test case 2 output

Richard Ricardo's Kung Fu Panda Po Power Level

This page will prompt Richard Ricardo for Kung Fu Panda Po's power level (in number). The program will convert the given input into Po's power metric.

less than 0: Po eats too much noodle  
0 to 200: Po reaches the training hall  
201 to 400: Po reaches the student barracks  
401 to 600: Po reaches the Peach Tree of Heavenly Wisdom  
601 to 800: Po climbs the Wu Da Mountains  
801 to 1000: Po trains at the Pool of Sacred Tears  
Greater than 1000: Po defeats Tai Lung

**Po's power level is 888.**

**Po trains at the Pool of Sacred Tears.**

[back to first page](#)

**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. 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>`.
4. Before adding PHP code, all html files must pass html validation at <http://validator.w3.org/> without any **error** (and with only 1 warning).
5. After adding PHP code, the generated html code (Firefox web browser > right-click > view page source) must also pass html validation at <http://validator.w3.org/> without any **error** (and with only 1 warning).
6. All css files must pass css validation at <http://jigsaw.w3.org/css-validator/> without any **error/warning**.
7. If your files do not pass the html and css validations, **2 points will be deducted for each html or css error/warning** found (1 warning allowed for html validator).
8. 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 Photoshop, or GIMP 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\_assignment1.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\_assignment1**) into a single zip file (or rar file) **lastname\_firstname\_assignment1.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\_assignment1\q1\index.htm
- lastname\_firstname\_assignment1\q1\index\_process.php
- lastname\_firstname\_assignment1\q1\style.css // you may put style.css in a subfolder
- lastname\_firstname\_assignment1\q2\index.htm
- lastname\_firstname\_assignment1\q2\index\_process.php
- lastname\_firstname\_assignment1\q2\style.css
- lastname\_firstname\_assignment1\q3\index.htm
- lastname\_firstname\_assignment1\q3\index\_process.php
- lastname\_firstname\_assignment1\q3\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**.