
CSCI 3300 Assignment 7

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, or same content, or same layout, or same color combination. If found, **both** students will get **0 points**.

Create a folder on your hard disk, name the folder **lastname_firstname_assignment7**. Save all the files from this assignment in this folder.

Create the following subfolders (in the folder lastname_firstname_assignment7): **q1, q2, q3, q4**.

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

- lastname_firstname_assignment7\q1\
- lastname_firstname_assignment7\q2\
- lastname_firstname_assignment7\q3\
- lastname_firstname_assignment7\q4\

Use http://www.javascriptlint.com/online_lint.php website to help debugging JavaScript. It will make your debugging process easier. All html (JavaScript) files must pass JavaScript validation at this website without any error, without any warning (**-2 points for each error, each warning**).

Total estimated time for this assignment: **12 hours**

Question 1 – JavaScript Chapter 7 (25 points)

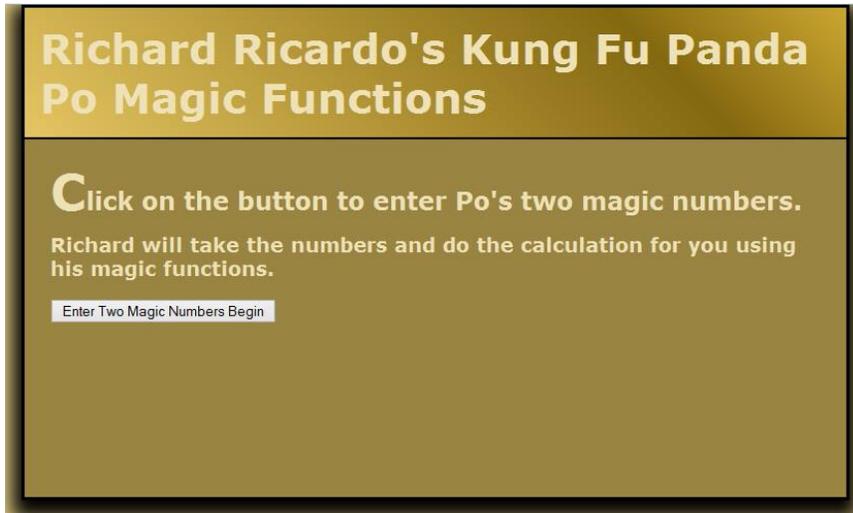
Estimated time: 3 hours

- Save question 1 files in subfolder “**lastname_firstname_assignment7\q1**”.
- Create a web page that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below.
- Create your page using “**<your name>**’s Richard Ricardo’s Kung Fu Panda Po Magic Functions” as the page title. Save the page as **index.htm**. Remember to document the html file with html comments.
- Allow the user to enter two magic numbers.
 - Make sure the numbers are **positive**
- The JavaScript code should include **four functions** as described below
 - The main function **RichardMath()** describe the selections, prompts user for two numbers (save as variables **x** and **y**), and **call the other three functions**
 - Function **RichardExp(x,y)** calculate the value x^y , and **return** the result
 - Function **RichardArea(x,y)** calculate the area of a right triangle (area = $\frac{1}{2} * \text{base} * \text{height}$), and **return** the result
 - Function **RichardDistance(x,y)** calculate the distance between two points (**x,y**) and **(0,0)**, and

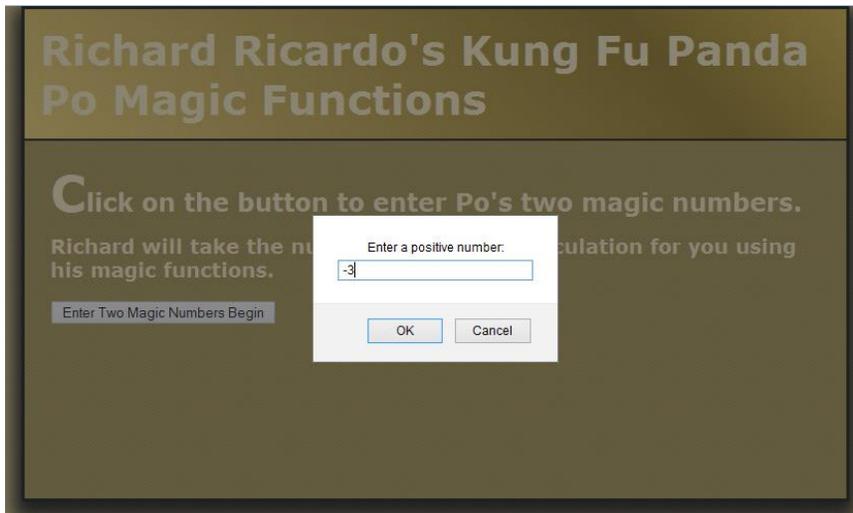
return the result. You may need the distance formula:
$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- Create a css file named **style.css** to format index.htm by creating your own layout (no two students should have the same layout). Use css comments to document the css program. You can use **the same (or similar) css file(s)** to format all questions.

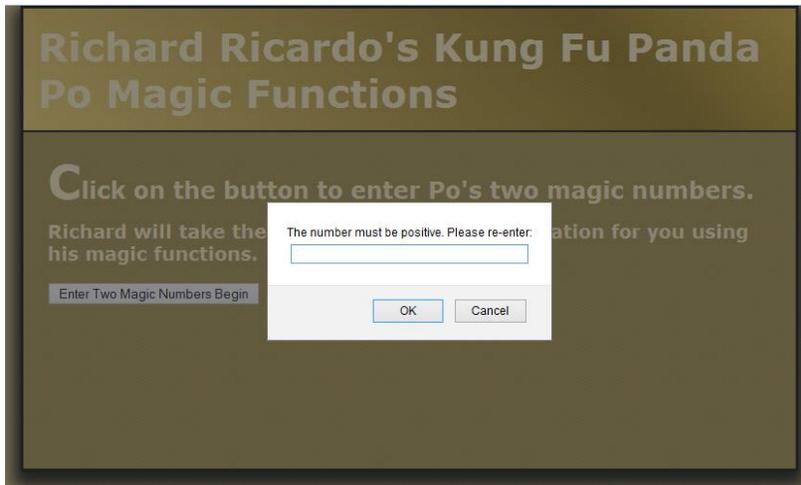
Example: Initial Page



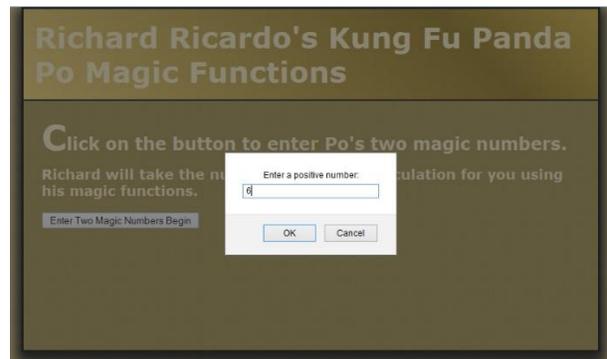
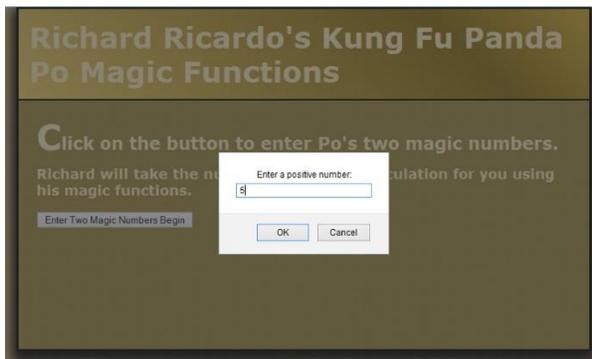
Example: Input q1a



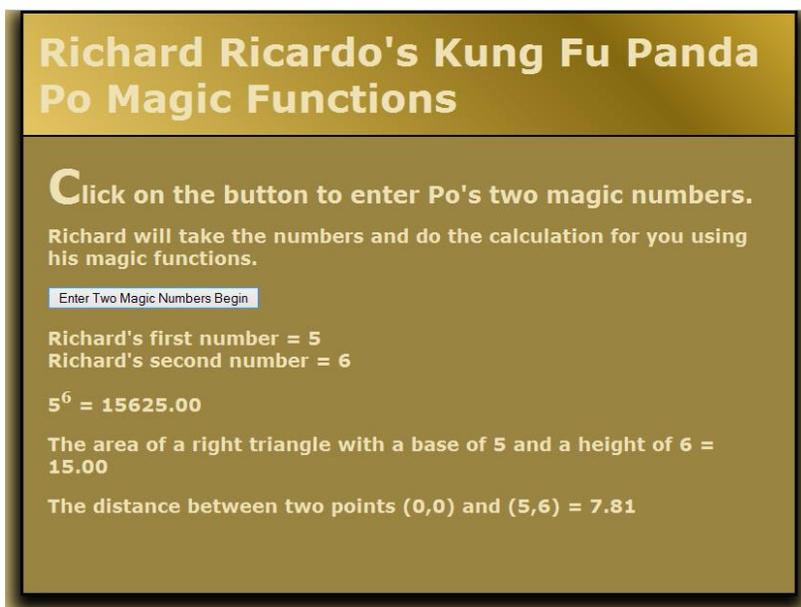
Example: Output q1a



Example: Input q1b



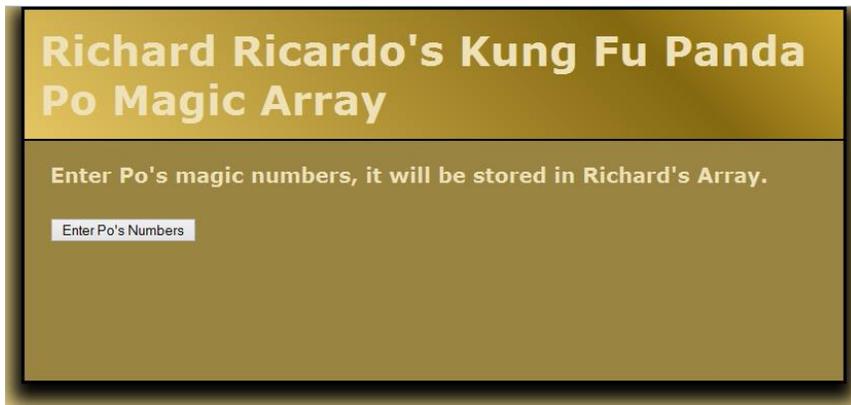
Example: Output q1b



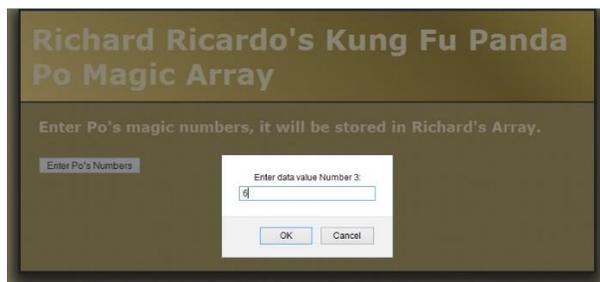
Question 2 – JavaScript Chapter 8 (25 points)**Estimated time: 3 hours**

- Save question 2 files in subfolder “**lastname_firstname_assignment7\q2**”.
- Create a web page that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below.
- Create your page using “**<your name>**’s Kung Fu Panda Po Magic Array” as the page title. Save the page as **index.htm**. Remember to document the html file with html comments.
- Write a program that asks the user for Po’s magic numbers.
 - The magic numbers must be stored in an **array**.
 - The program needs to calculate the **mean (average)** of the numbers (from the **array**).
 - The program needs to count the number of magic numbers that are **bigger than the mean** number (from the **array**).
 - The program needs to count the number of magic numbers that are **smaller than the mean** number (from the **array**).
- Create a css file named **style.css** to format index.htm by creating your own layout (no two students should have the same layout). Use css comments to document the css program.

Example: Initial Page

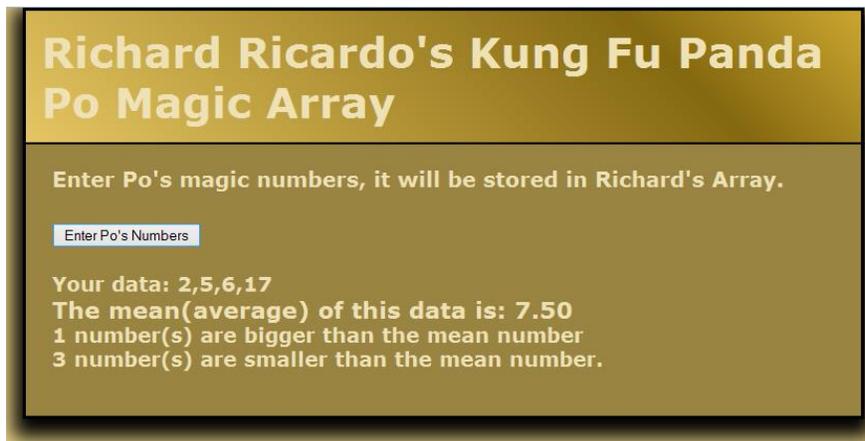


Example: Input q2





Example: Output q2



Question 3 – JavaScript Chapter 8 (25 points)

Estimated time: 3 hours

- Save question 3 files in subfolder “**lastname_firstname_assignment7\q3**”.
- Create a web page that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below.
- Create your page using “<your name>’s Kung Fu Panda Po Magic Square” as the page title. Save the page as **index.htm**. Remember to document the html file with html comments.
- Write a program that allows the user to enter Po’s magic numbers.
 - Po’s magic square is a **two-dimensional array** of **positive integers** in which the following is true:
 - The number of rows equals the number of columns.
 - **Every row, column, and the two diagonals** add up to the same number.
 - The web page must have a table that has **four rows and four columns**.
 - Allow the user to enter values for each cell.
 - Store the values in a **two-dimensional array** named **RichardMagic** and determine if it is Po’s magic square.
- Create a css file named **style.css** to format index.htm by creating your own layout (no two students should have the same layout). Use css comments to document the css program.

Example: Initial Page

Richard Ricardo's Kung Fu Panda Po Magic Square

Richard will check if the numbers form a magic square. A magic square exists when all the rows, columns, and diagonals of a square add up to the same number.

Enter Po's magic numbers

Check if it is a magic square

Result:

Example: Input q3a

Richard Ricardo's Kung Fu Panda Po Magic Square

Richard will check if the numbers form a magic square. A magic square exists when all the rows, columns, and diagonals of a square add up to the same number.

Enter Po's magic numbers

Enter a number for row 1, column 1:

OK Cancel

Check if it is a magic square

Result:

Richard Ricardo's Kung Fu Panda Po Magic Square

Richard will check if the numbers form a magic square. A magic square exists when all the rows, columns, and diagonals of a square add up to the same number.

Enter Po's magic numbers

Enter a number for row 1, column 2:

OK Cancel

1			

Check if it is a magic square

Result:

Skip screen captures...

Enter all 16 numbers as shown in the output screen.

Example: Output q3a

Richard Ricardo's Kung Fu Panda Po Magic Square

Richard will check if the numbers form a magic square. A magic square exists when all the rows, columns, and diagonals of a square add up to the same number.

Enter Po's magic numbers

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Check if it is a magic square

Result: Sorry, your square is not a magic square

Example: Input q3b

Richard Ricardo's Kung Fu Panda Po Magic Square

Richard will check if the numbers form a magic square. A magic square exists when all the rows, columns, and diagonals of a square add up to the same number.

Enter Po's magic numbers

Enter a number for row 1, column 1:

16			

Check if it is a magic square

Result:

Richard Ricardo's Kung Fu Panda Po Magic Square

Richard will check if the numbers form a magic square. A magic square exists when all the rows, columns, and diagonals of a square add up to the same number.

Enter Po's magic numbers

Enter a number for row 1, column 2:

16	2		

Check if it is a magic square

Result:

Skip screen captures...

Enter all 16 numbers as shown in the output screen.

Example: Output q3b

Richard Ricardo's Kung Fu Panda Po Magic Square

Richard will check if the numbers form a magic square. A magic square exists when all the rows, columns, and diagonals of a square add up to the same number.

Enter Po's magic numbers

16	2	3	13
5	11	10	8
9	7	6	12
4	14	15	1

Check if it is a magic square

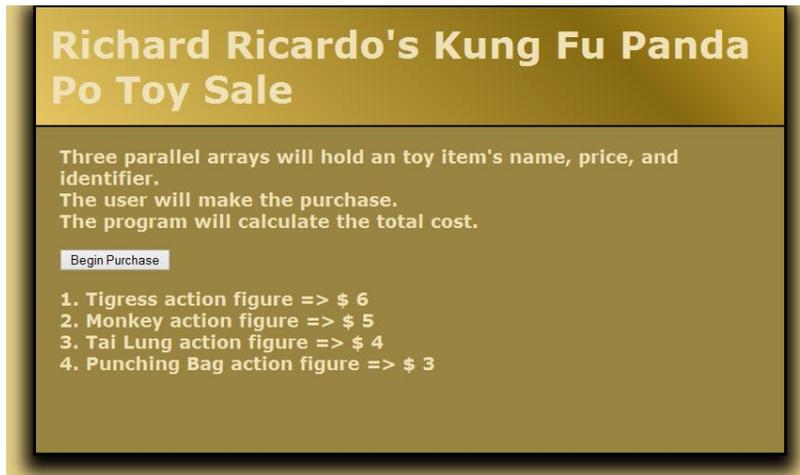
Result: Wow! This is a magic square

Question 4 – JavaScript Chapter 8 (25 points)

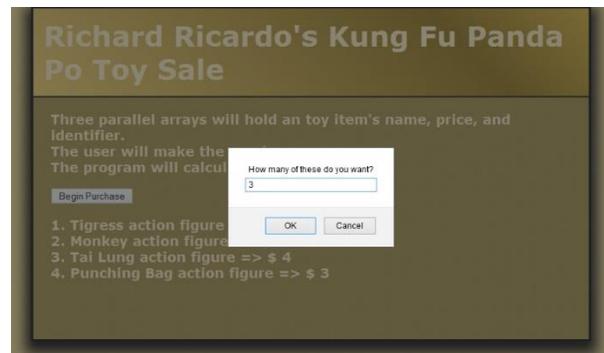
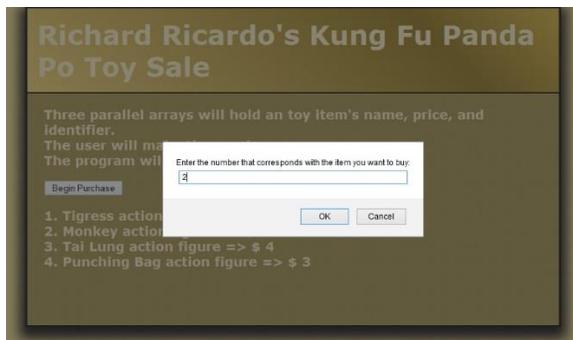
Estimated time: 3 hours

- Save question 4 files in subfolder “**lastname_firstname_assignment7\q4**”.
- Create a web page that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below.
- Create your page using “<your name>’s Kung Fu Panda Po Toy Sale” as the page title. Save the page as **index.htm**. Remember to document the html file with html comments.
- Allow the user to buy Po’s toys as listed on the page.
 - The page should include **parallel arrays** (at least two).
 - One **array (RichardItem)** holds the **names** of the toys.
 - `var RichardItem = new Array("Tigress action figure", "Monkey action figure", "Tai Lung action figure", "Punching Bag action figure");`
 - One **array (RichardPrice)** holds the corresponding **prices** for each toy.
 - `var RichardPrice = new Array(6, 5, 4, 3);`
 - Retrieve values from the **parallel arrays**, display the toys’ names/prices as shown (hint: onload).
 - The user should be prompted to select a toy and a quantity.
 - The page will display the total cost for that quantity of the toy chosen (by using the values from the **parallel arrays**).
- Create a css file named **style.css** to format index.htm by creating your own layout (no two students should have the same layout). Use css comments to document the css program. You can use **the same (or similar) css file(s)** to format all questions.

Example: Initial Page

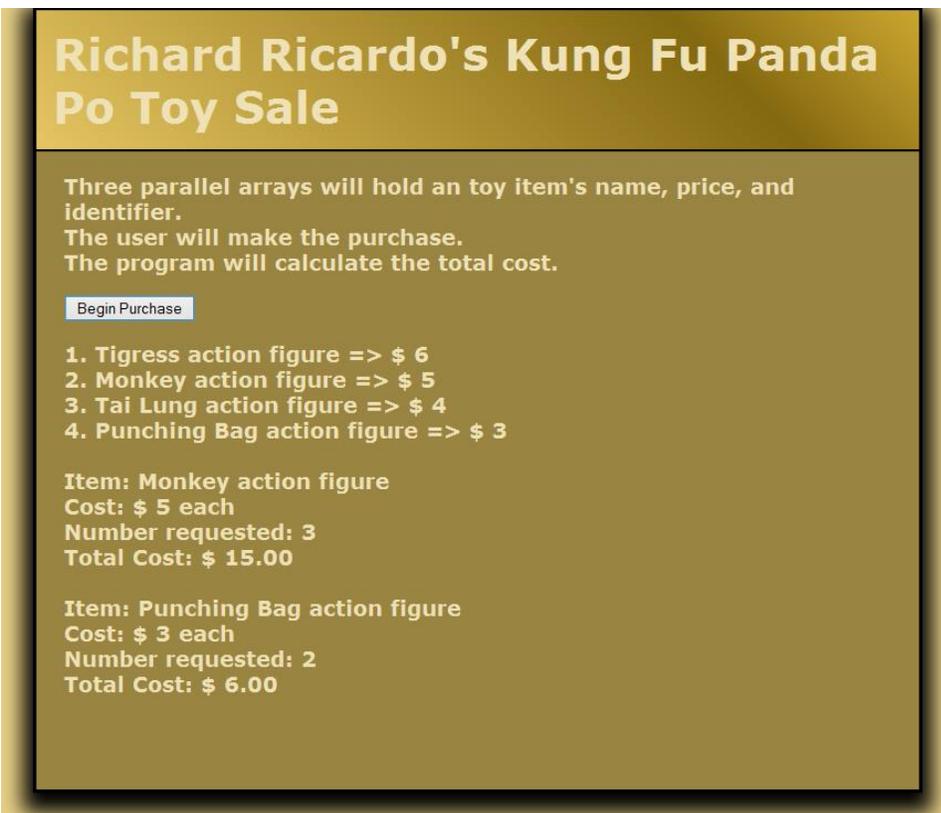


Example: Input q4





Example: Output q4



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, or same content, or same layout, or same color combination. If found, both students will get **0 points**.
3. 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.
4. All css files must pass css validation at <http://jigsaw.w3.org/css-validator/> without any **error**.
5. All html(JavaScript) files must pass JavaScript validation at http://www.javascriptlint.com/online_lint.php without any **error**, without any **warning**.
6. If your html file contains any css component, your html file must pass both html validation (3 above), and css validation (4 above) without any error.
7. If your files do not pass the validations, **2 points will be deducted** for **each error** (and each JavaScript warning) found.
8. Document (comment) your html files (`<!-- -->`), css files (`/* */`), and JavaScript files (`/* */`).

Submission instructions:

- You need to test the above document(s) in your web browser.
- Do screen capture(s) of the **initial page** 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.
- For this assignment, you only need to do screen capture(s) of the **initial page** and the related **output(s)**, you do not need to do screen capture(s) of the input pages. Provide **2 different test cases** for each question. In other words, for **each question**, you need to have **1 initial page** screen capture and **2 related output(s)** screen captures.
- Do screen capture(s) of html validation results and css validation results, cut and paste them into the word document. You do not need to do screen capture(s) of JavaScript Lint validation results.
- 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 output (all html 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_assignment07.pdf**.
2. All html file(s), css file(s), and other related files (e.g. image files). Zip your file folder (lastname_firstname_assignment7) into a single zip file (or rar file) **lastname_firstname_assignment07.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_assignment7\q1\index.htm
- lastname_firstname_assignment7\q1\style.css
- lastname_firstname_assignment7\q2\index.htm
- lastname_firstname_assignment7\q2\style.css // you may put style.css in a subfolder
- lastname_firstname_assignment7\q3\index.htm
- lastname_firstname_assignment7\q3\style.css
- lastname_firstname_assignment7\q4\index.htm
- lastname_firstname_assignment7\q4\style.css

Please submit an electronic copy (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**.