CSCI 3300 Assignment 3

Total estimated time for this assignment: **10 hours**

When you see "Richard Ricardo" in the example screen captures, change it to **<your name>**. (0 points if not)
When you see "Richard" in the example screen captures, change it to **<your first name>**. (0 points if not)
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_assignment3.** Save all the files from
this assignment in this folder.

Create the following subfolders (in the folder lastname_firstname_assignment3)**: q1, q2, q3, q4, q5**.
As a result, you should have the following folder (directory) structure for this assignment: (-2 points if wrong)
- lastname_firstname_assignment3\q1\
- lastname_firstname_assignment3\q2\
- lastname_firstname_assignment3\q3\
- lastname_firstname_assignment3\q4\
- lastname_firstname_assignment3\q5\

Use Firefox **Web Console** (Tools > Web Developer > Web Console) to help debugging JavaScript. All html
(JavaScript) files must pass JavaScript validation at Firefox Web Console without any error, without any
warning **(-2 points for each error, each warning)**.
- Turn on your Firefox **Menu Bar** (Hint: right-click).
- Make sure your Firefox is up to date (Help > About Firefox).
- Turn on Web Console when you code (Tools > Web Developer > Web Console).
- Fix any error/warning immediately when you see them. Do NOT accumulate errors.

When you view page source in a web browser, `<!DOCTYPE html>` must be at the top of every page. In
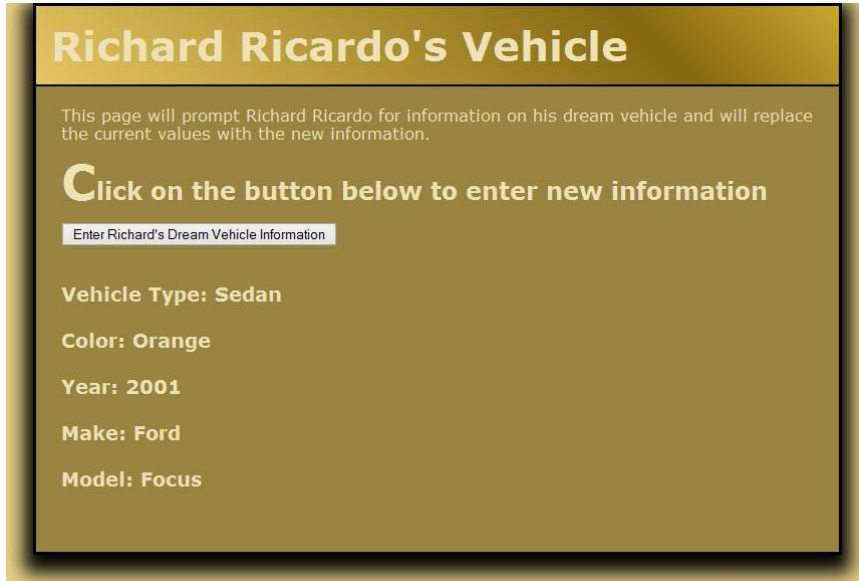other words, all pages must be written in HTML5. **(-20 points** if not)

Question 1 – JavaScript Chapter 1 (20 points)   **Estimated time: 2 hours**
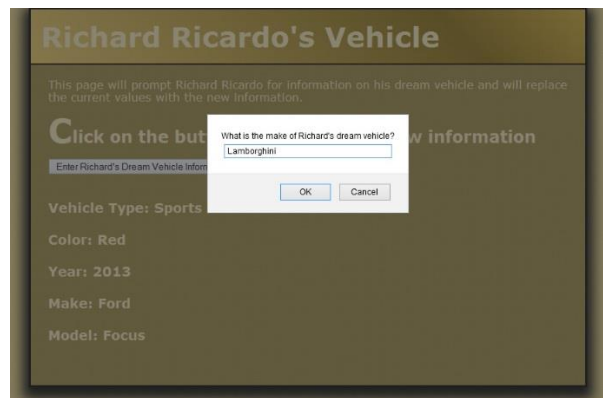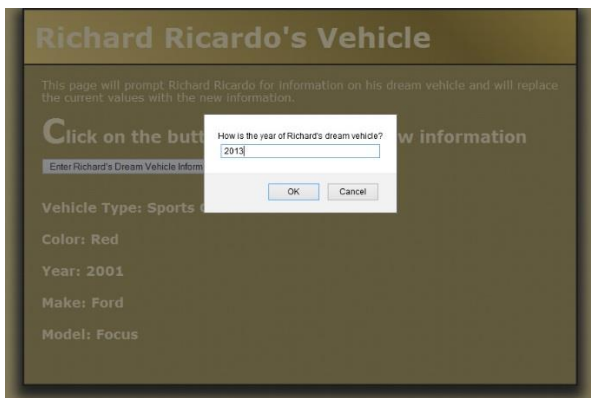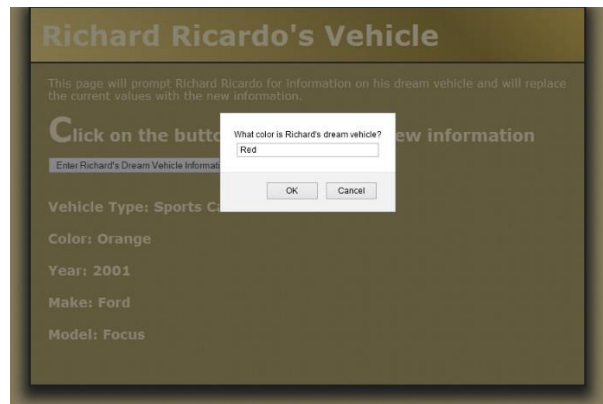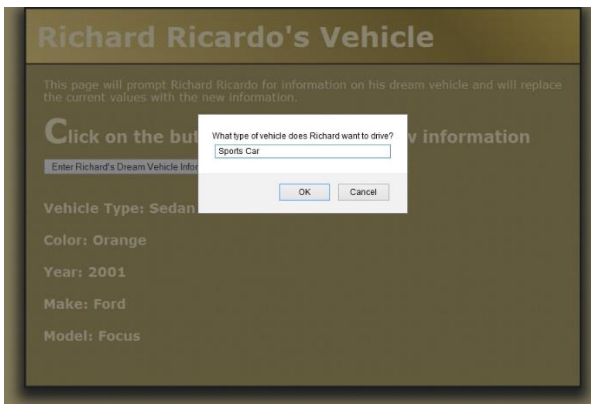- Save question 1 files in subfolder "**lastname_firstname_assignment3\q1**". (-15 points if no JavaScript)
- Create a web page that displays the default information (example shown) about **your vehicle**. If you do
  not have a vehicle, display the information about the first vehicle you can afford to buy.   (1 point)
- The initial page should look like the one shown below.   (1 point each, total 3 points)
- **Prompt** the user to **enter the information** of his/her dream vehicle, and use that to replace the defaults.
  (input as shown: 1 point each, total 5 points; output as shown: (1 point each, total 5 points)
- Create your vehicle page using "**<your name>**'s vehicle page**"** as the page title. Save the page as
  index.htm. Document the html file with html and JavaScript comments. (1 point each, total 3 points)
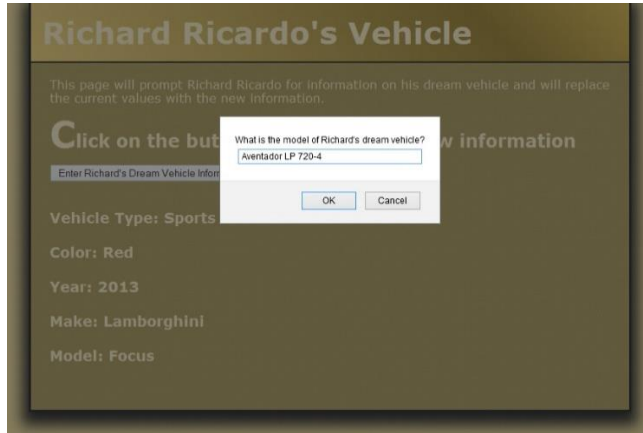
- Create a css file named **style.css** to format index.htm by creating your own layout (no two students should have the same layout). Use css comments to document the css program. You can use **the same (or similar) css file(s)** to format all questions. (1 point each, total 2 points)
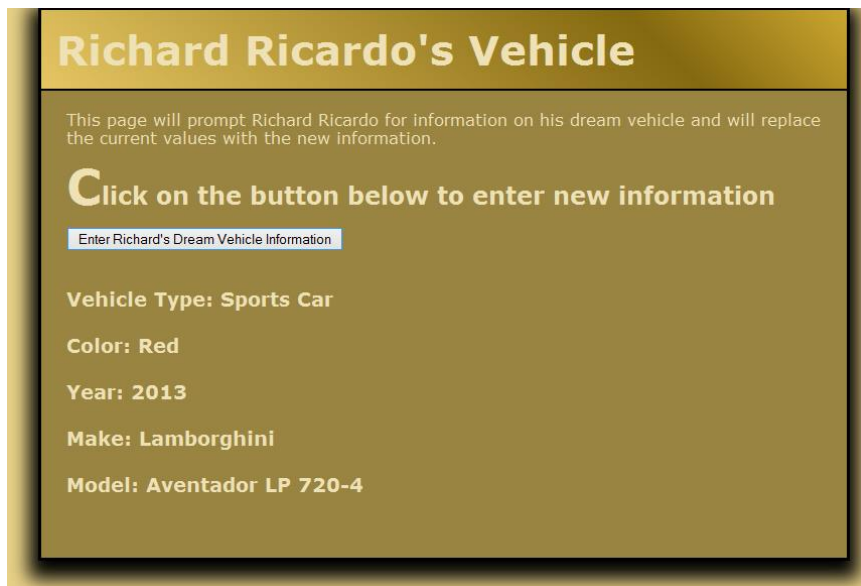
Example: Initial Page

Example: Input

Example: Output

Question 2 – JavaScript Chapter 1 (20 points)   **Estimated time: 2 hours**

- Save question 2 files in subfolder "**lastname_firstname_assignment3\q2**". (-15 points if no JavaScript)
- Create a web page that contains **two buttons (heading tags with border)** as shown. (2 points each, total 4)
- When the first one is clicked, the user will be prompted to enter his/her favorite soccer team's name. (1 point)
- When the second one is clicked, the user will be prompted to enter his/her favorite football team's name. (1 point)
- The page will then be **updated** as shown.   (2 points each, total 4 points)
- Create your page using "**<your name>**'s Favorite Soccer & Football Team**"** as the page title. Save the page as **index.htm**. Document the html file with html & JavaScript comments. (1 point each, total 3 points)
- 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. (1 point each, total 2 points)

Example: Initial Page

Example: Input

Example: Output

Question 3 – JavaScript Chapter 2 (20 points)   **Estimated time: 2 hours**

- Save question 3 files in subfolder "**lastname_firstname_assignment3\q3**". (-15 points if no JavaScript)
- Create a web page that allows a user to create a **soccer school name** as shown. (1 point each, total 3 points)
- The use should be prompted to enter his or her **first name**, **last name**, and the **school's name**. (1 point each, total 3 points)
- The program should create a soccer school name that consists of the user's **initials** concatenated with the **school name**, plus "**School**", as shown.   (1 point each, total 3 points)
- The page will then be **updated** as shown.   (1 point)
- Create your page using "**<your name>**'s Soccer School Name**"** as the page title. Save the page as **index.htm**. Document the html file with html and JavaScript comments. (1 point each, total 3 points)
- 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. (1 point each, total 2 points)

Example: Initial Page

Example: Input

Example: Output

Question 4 – JavaScript Chapter 2 (20 points)   **Estimated time: 2 hours**
- Save question 4 files in subfolder "**lastname_firstname_assignment3\q4**". (-15 points if no JavaScript)
- Create a web page that displays the cost of a soccer ticket as shown.   (1 point each, total 2 points)
- The user should be prompted to enter his or her **age**.   (1 point)
- The output should be a message telling the user what the customer's soccer ticket will cost, based on the following criteria:
  - Under age 5 (age < 5), cost is $2   (2 points)
  - Between ages 5 and 15, cost is $6   (2 points)
  - Older than 15 (age > 15), cost is $8   (2 points)
- The page will then be **updated** as shown.   (2 points)
- Create your page using "**<your name>**'s Soccer Ticket Price**"** as the page title. Save the page as **index.htm**. Document the html file with html and JavaScript comments. (1 point each, total 3 points)
- 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. (1 point each, total 2 points)

Example: Initial Page

Example: Input

Example: Output

Question 5 – JavaScript Chapter 2 (20 points)   **Estimated time: 3 hours**
- Save question 5 files in subfolder "**lastname_firstname_assignment3\q5**". (-15 points if no JavaScript)
- Create a web page that contains **two buttons (heading tags with border)** as shown. (2 points each, total 4)
- The web page should check to see if the user enters an age within a given age range.
- The use should be prompted to enter his or her **age**.   (1 point)
- Write a program that uses two different functions/methods to test if an input is between 15 and 21
  (including 15 and 21).   (1 point for each function total 2)
- One function/method must use the AND operator (&&).   (3 points)
- One function/method must use the OR operator (||).   (3 points)
- The web page should display the two equivalent expressions.
- The results will then be **displayed** as shown.   (1 point)
- Create your page using "**<your name>**'s Age Range Check**"** as the page title. Save the page as **index.htm**.
  Document the html file with html and JavaScript comments.   (1 point each, total 3 points)
- 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. (1 point each, total 2 points)

Example: Initial Page

Example: Input
Note: Required input is the **same** for both buttons (heading tags with border)

Example: Output
Note: Output is the **same** for both buttons (heading tags with border), the difference is one function uses AND, one function uses OR, to check the range.

If user enters 15, output is shown above.   If user enters 23, output is shown above.

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. 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)
4. All html files must pass html validation at http://validator.w3.org/ without any **error/warning** (with only 2 warnings). Use the validator's "File Upload" tab to check each file. **(-2 points** for each error/warning, **2 warnings allowed**)
5. All css files must pass css validation at http://jigsaw.w3.org/css-validator/ without any **error/warning**. **(-2 points** for each error/warning)
6. All html (JavaScript) files must pass JavaScript validation at Firefox **Web Console** (Tools > Web Developer > Web Console) without any **error**, without any **warning**. **(-2 points** for each error/warning)
7. 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.
8. If your files do not pass the validations, **2 points will be deducted** for **each error / warning** found.
9. Document (comment) your html files `(<!-- -->)`, css files `(/* */)`, and JavaScript files `(/* */)`. **(-1 point** for each file with insufficient comments)

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. (-50 points for no test cases, -5 points for only 1 test case)
- 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 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 and css validation results, name the file **lastname_firstname_assignment03.pdf**.
2. All html file(s), css file(s), and other related files (e.g. image files). Zip your file folder (lastname_firstname_assignment3) into a single zip file (or rar file) **lastname_firstname_assignment03.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_assignment3\q1\index.htm
- lastname_firstname_assignment3\q1\style.css
- lastname_firstname_assignment3\q2\index.htm
- lastname_firstname_assignment3\q2\style.css   // you may put style.css in a subfolder
- lastname_firstname_assignment3\q3\index.htm
- lastname_firstname_assignment3\q3\style.css
- lastname_firstname_assignment3\q4\index.htm
- lastname_firstname_assignment3\q4\style.css
- lastname_firstname_assignment3\q5\index.htm
- lastname_firstname_assignment3\q5\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**.