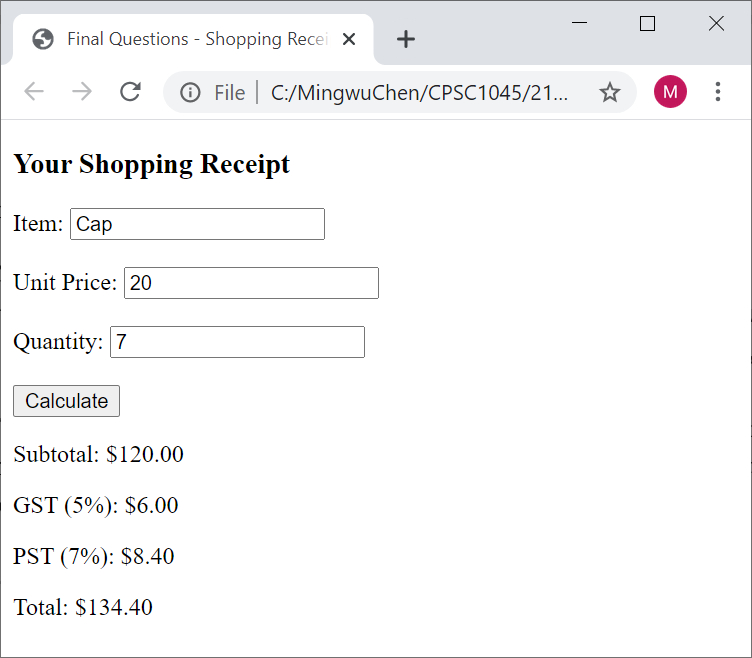
# CPSC 2030 – Web Development II

**Lab4: [45 marks] JavaScript Fundamentals**

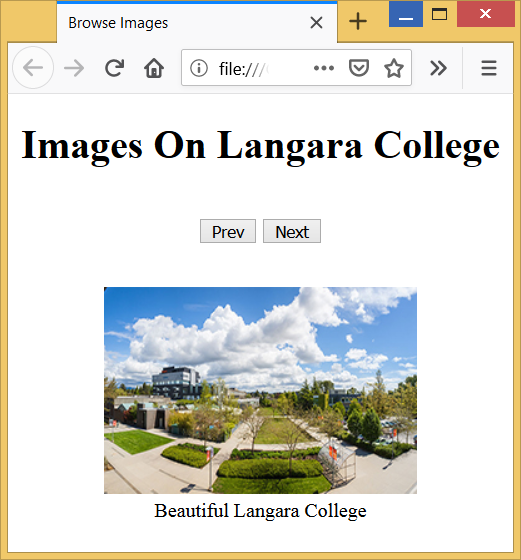
|  |
| --- |
| **Objectives** |
| * Practice on the fundamentals of JavaScript * Create and call functions * Work with JavaScript objects |
| **Preparation** |
| Create a new folder named **Lab4** under your CPSC2030\Labs folder. |
| **What to do** |

1. [10 marks] Shopping Receipt (**shoppingreceipt.html**)

Write JavaScript code (**internal or external**) to let the user enter the Item name, Unit Price, and Quantity for the item. The price for every third piece of the same item is half of the regular price. When the button Calculate is clicked, all the four values are calculated and displayed as shown on the following sample page (keep two decimals for all the calculated values).



1. [15 marks] Image Browsing(filename: **imagebrowsing.html**)
2. Download the **LangaraImages.zip** file from BrightSpace.
3. Create a webpage like the following:



1. The HTML content of the page is like the following:

<html lang="en">

<head>

<title>Browse Images</title>

<meta charset="utf-8">

</head>

<body>

<div style="text-align:center">

<h1>Images On Langara College</h1><br>

<button id="prev" onclick="prevImage()">Prev</button>

<button id="next" onclick="nextImage()">Next</button><br><br>

<figure>

<img id="langara" src="LangaraImages/langara.jpg" width="250" height="166">

<figcaption id="caption">

Beautiful Langara College

</figcaption>

</figure>

</div>

</body>

</html>

1. Write your JavaScript (**internal or external)** todo the following
   1. Put all the images into an array
   2. Create a second array to store the corresponding captions for the images
   3. Implement the two button click event handlers to handle the next and prev functions.
   4. When clicking the next button reaches the last image, it should wrap around to the first image. Similarly, for the prev button, it should wrap around to the last image.
2. [20 marks] Zoo Species(filename: **zoospecies.html and zoospecies.js**)
3. Download **zoospecies.zip** (unzip it to zoospecies.html) and **zoospecies.js** from BrightSpace. The HTML file should not be edited.
4. Inside the JavaScript file create a **constructor** that creates species objects. A species object should store the following properties:
   * Name - the name of the species
   * Count - the number of animals of this species in the zoo.
5. Create a global array that store an unknown number of animal objects.
6. Write the addSpecies() function that runs when the Add Animal button is clicked. This function should do the following things:
   * Determine the species and count of the animal from the user input fields
   * Create a species object from the user information
   * Push the species object onto the array if it does not exist, otherwise just add the count to the existing species.
   * Update the output/display
7. Complete the createList() function which fills in the unordered list for the zoo
8. Complete the removeSpecies() function which removes a species with the given count and then updates the zoo list. If the new count for this species becomes zero or negative, remove this species from the zoo list.

|  |
| --- |
| **What to hand in** |
| * Zip your Lab4 folder and upload it to BrightSpace. |
| **When to hand in** |
| * By 11:59pm, Sunday, October 2, 2022 |