Assignment 1

Client-side programming with Javascript

Submit a single zip file called **assignment1.zip**.
This assignment has 100 marks.
You should read the marking scheme posted on cuLearn for details.

Assignment Background

In this assignment, you will develop a web page that allows a user to browse menus for several restaurants, add items from a restaurant to an order, and simulate placing an order. Everything you implement for this assignment will involve client-side programming within the browser. You must implement everything using plain Javascript and cannot use external libraries/tools such as jQuery and Bootstrap.

Before starting your design for the assignment, you are encouraged to read through the entire specification. It is possible that later problems will inform your design decisions and, by preparing in advance, you can avoid having to significantly refactor your code as you progress through the assignment.

To begin the assignment, download the `client.js` file from cuLearn. This file contains three variables (`aragorn`, `legolas`, `frodo`) containing the menu data for each of the three included restaurants. There is an also an array called `restaurants` which contains each of the three menu objects. Each menu object contains four properties:

1. `name` – the name of the restaurant
2. `min_order` – the minimum order amount required to place an order
3. `delivery_charge` – the delivery charge for this restaurant
4. `menu` – an object containing the menu data for this restaurant

The menu object may have any number of properties, each of which indicates a category within the restaurant's menu. The value associated with each category property will also be an object. The properties of the category object are unique IDs associated with each menu item. The value associated with each unique menu item ID includes the name, description, and price of that item. A general example of the format of a restaurant's data is included in the `example-menu-data.txt` file posted on cuLearn. You may use the data as it is provided, or you may re-format the data in any way you desire.

Page Load (10 marks)

Create an HTML page called `order.html`. When the page loads, the following requirement must be met:

1. The page should contain a drop-down list or other method for selecting the current restaurant. You can either have a default restaurant selected, in which case the page contents should be initialized to the menu of that restaurant, or have a 'Select a restaurant...' type of option with an initially blank menu section.
2. The contents of the drop-down list must be dynamically created from the provided restaurant data. You should not hard code the selections.

Selecting a Restaurant (20 marks)

When the selected restaurant is changed, the following requirements must be met:

1. If there is a current order with items in it, a prompt should be displayed to the user to confirm whether they want to clear the order or not. If the user does not want to clear the order, then no data on the page should change (i.e., it cancels the affect of the change in restaurant).
2. If the user accepts clearing the current order, or if there is no current order, then the newly selected restaurants information should be placed on the page. Any information from the previous restaurant should be removed. Any existing order information should be cleared.
3. The restaurant's name, minimum order, and delivery fee should be displayed near the top of the page.
4. The menu and order information should be displayed in a three-column layout below the basic restaurant information. The left column must have a list of the categories in the restaurant's menu. These categories should also be links to allow the user to skip directly to that category within the menu. The middle column should present the menu of the restaurant (see Menu Requirements below). The right column should present the current order summary (see Order Summary Requirements below).

Menu Requirements (20 marks)

1. The menu should be organized so each category of item is clearly separated.
2. Each item should be clearly separated (e.g., using a list) and show the name, description, and price of the item.
3. The add.jpg image should be shown near each item and clicking this image should allow the user to increase the quantity of that item in the order by 1.

You will have to scale the size of the image. You can use an alternate image or element, but it must be clear that it is responsible for increasing the amount of an item in the order.

4. All dollar values should be displayed to two decimal places.

Order Summary Requirements (30 marks)

1. For each item currently in the order, the summary should include the item's name, number of units in the order, and total price of that item (unit price * number of units).
2. The remove.jpg image should be shown near each item and clicking this image should allow the user to decrease the quantity of that item by 1. If the quantity goes to 0, the item should not show up in the summary. You may use an alternate image or element, but it must be clear that it is responsible for decreasing the amount of an item in the order.
3. Following the list of items, the current subtotal (sum of all item prices) should be displayed, along with the delivery fee, the tax (use a tax rate of 10%) and the current order total.
4. If the current subtotal is greater than or equal to the minimum order amount of the selected restaurant, there should be an element that allows the user to submit the order. When this element is clicked, an alert should be displayed confirming that the order has been submitted and the page should be reset.
5. If the current subtotal is less than the minimum order, a small message should be placed at the bottom of the summary indicating the total amount of dollars that must be added before the order can be placed. For example, if the minimum order is $25 and the current subtotal is $10, the message could say "You must add $15.00 more to your order before submitting".
6. All dollar values should be displayed to two decimal places.

Overall Page Quality (10 marks)

These marks will be allocated for the overall quality of your page's implementation. This will include factors such as the visual appeal of your page, as well as the responsiveness of your page to user actions and changes. For example, how does your page react to the changing of the browser dimensions? Is information easy to find on your page? Is everything on the page formatted in a clear way? If you struggle to make visually appealing pages, you can focus on adding behavioural elements to your page (e.g., aligning the categories and order summary with the current location in the menu). You are also free to add additional components or functionality to your page. Include a description of any additions/changes you made within your README.txt file.

Code Quality and Documentation (15 marks)

Your code should be well-written and easy to understand. This includes providing clear documentation explaining the purpose and function of pieces of your code. You should use good variable/function names that make your code easier to read. You should do your best to avoid unnecessary computation and ensure that your code runs smoothly throughout operation. You should also include a README.txt file that explains any design decisions that you made, as well as any additional instructions that may be helpful to the TA.

Recap

Your zip file should contain all the resources required for your assignment to run. The TA should be able to open the order.html page and use your page without any additional setup.

Submit your **assignment1.zip** file to cuLearn.

Make sure you download the zip after submitting and verify the file contents.