LAB – Javascript

Question 1

Write a regular expression to match a price. A price always starts with a dollar sign. Any amount of numbers can come before the decimal. Two numbers should always follow the decimal.

Valid:	Invalid:
$14.99	$14
$1234567.00	$134213.89money
$.90	$1.1a

Question 2

Write a regular expression to match a self closing HTML tag. A tag must start with “<” and end with “/>”. It must have a name and can have 0, 1 or more properties. You do not need to wory about white space inside the tag

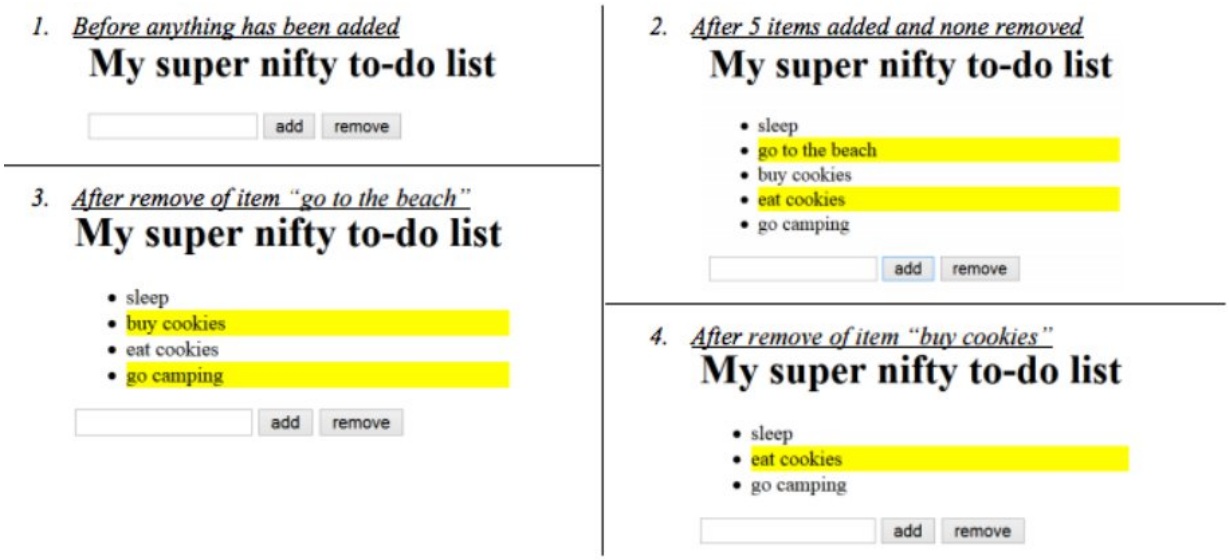
Valid:	Invalid:
	</>
<img />	
<a href=”foo.html” id=”stuff” />	

Question 3

Write JavaScript code to add behavior to the following page for keeping track of a to-do-list. The page UI allows the user to type an item into a text box. The user can click the "add" button to add the item to the bottom of the list. Each word in the phrase should be inserted as a li, inside a ul with the id of list. If the user wishes to remove an item he or she can type the text of the item he or she wishes to remove in the text box and click the "remove" button. This should be case insensitive. For example, if the list only contains "foo" and the user tries to remove "FoO", it should be removed. If the user tries to remove an item that is in the list multiple times only the first occurrence should be removed. The items should have background colors that alternate between white and yellow (first white, then yellow, then white, yellow, etc.). This should still be the case no matter how many items are removed or added and no matter what order these operations are done in. You may not use the CSS3 nth-child pseudo selector to do this. The code should work for multiple clicks of the buttons. On each click it should clear any previous information you typed in the input boxes. Do not use any JavaScript libraries such as jQuery or Prototype. Here is the relevant HTML code for the page:

```
<h1>My super nifty to-do list</h1>
<ul id="list"></ul>
<div>
<input type="text" id="item" />
<button id="add">add</button>
<button id="remove">remove</button>
</div>
```

These screenshots show the state after items have been added, and the state after items have been removed.



Question 4

Use this html document for reference for the following 3 problems.

```
<!DOCTYPE html>
<html lang="en">
<head>
<script src="sesame-street.js" type="text/javascript"></script>
<script src="fetch-pie.js" type="text/javascript"></script>
</head>
<body>
<h1>Whitaker's Desserts</h1>
<h2 id="cookie-header">Whitaker's Cookie Jar:</h2>
<ul id="cookie-jar">
<li class="cookie">Chocolate Chip</li>
<li class="cookie">Oatmeal raisin</li>
<li class="cookie">Macaroon</li>
</ul>
<p id="cookie-count"></p>
<h2>Whitaker's Pie Cupboard:</h2>
<ul id="pie-cupboard">
</ul>
<button id="moar-pie">Moar Pie!</button>
</body>
</html>
```

Implement sesame-street.js

a. (Big Bird Yellow): When the page loads, apply a style to the #cookie-header <h2> that sets

b. (Count Chocula): Implement javascript in sesame-street.js such that when the page loads, you find all of the cookie list items in the cookie-jar, count them, and set the text of the #cookie-count <p> to be. <# of cookies>! There are <# of cookies> cookie(s) in the cookie jar! the text color to hex value #f7f16d

   (Replace '<# of cookies>' with the correct number).

c. (Cookie Monster hungry): Implement javascript logic to remove the last cookie in Whitaker's cookie jar every 30 seconds. Make sure to update the #cookie-count when you remove a cookie: to accomplish this, you may assume that your solution from part(b) is correct and functional.