

University of Washington, CSE 154

Section 10: Practice Final Exam Problems

section problems by Sylvia Tashev, Brian Le, and several UW student volunteers

Today we will solve some practice problems similar to the problems that will be on our final exam. Try these out first using paper and pencil!

1. HTML / CSS Interpretation

Draw a picture of how the following HTML and CSS code will look when the browser renders it onscreen. Indicate a background coloring by shading lightly or by drawing repeated diagonal lines like this. Draw the image `geneva.jpg` as a smiley face occupying roughly 20% of the page width.

HTML

```
<h1>Baby Geneva's Web Page</h1>
<div id="notice">
  
  What is your name?
  <input type="text" size="20" value="Geneva" />
  <span class="notice">(Mine is cuter!)</span>
</div>
<p class="notice">I <br /> am <br /> 15 <br /> months <br /> old</p>
<p class="notice">
  My <br /> favorite <br /> toy <br /> is <br /> your <br /> cell phone
</p>
<h2>(written by Geneva, May 2008)</h2>
```

CSS

```
h1, .notice {
  text-align: center;
  border: 2px solid black;
}
h2 { clear: both; }
h2, #notice {
  border-top: 1px dashed black;
  border-bottom: 1px dashed black;
}
p.notice {
  border: 2px solid black;
  float: right;
  width: 10%;
}
#notice .notice {
  text-decoration: underline;
}
```

2. HTML / CSS - Apples:

Write HTML (just what's in the body) and CSS code to reproduce the following page:

Apples

"An apple a day keeps the doctor away!"



Some types of apples:

- Red delicious
- Golden delicious
- Granny Smith
- Crabapple
- Gala apple
- Red Rome apple
- Ginger gold
- Fuji apple

Other apples:

Apple iPod

This apple is poisonous.

Steve Jobs

The tallest apple.

The following are details about the appearance of the page:

- Make all headings green, and make them use Garamond font or any serif font on the system.
- The quote should be italicized.
- The apples picture name is `greenapples.jpg`. It also links to <http://www.apple.com/>.
- All pictures should have no borders and be 250 pixels wide.
- The right section has a 2 pixel wide outset green border and a background of `#99cc99`.
- Every other line in the "types of apples" list is white and its font-size is 120% as large as the other list items.
- The definition terms are bolded.
- Notice that there are two distinct sections on the page.
- The content inside each section should be 20 pixels away from all edges.
- Note: Don't forget to use proper tags for all of the elements that clearly describe the content on the page.

3. PHP - Image Gallery Search:

The following HTML snippet is the skeleton of a simple search page for an image gallery.

```
<h1>Image Gallery Search</h1>
<form action="search.php" method="get">
  <fieldset>
    Type a query:
    <input type="text" name="query" /> <br />
    <input type="submit" value="Search" />
  </fieldset>
</form>
```

Write a PHP program called `search.php` to implement the searching feature. An image is considered a "match" to the search string if the name of the image contains the entirety of the search string. For example, a query of "tea" might match "**tea**.jpg" or "**steam**boat.jpg".


You should output an unordered list of links to all matches. A blank query should return no results. The gallery stores all of its images in a directory called `images`. You may assume there are only image files in the `images` directory. The search should be case-insensitive and you should eliminate the whitespace surrounding a query before processing it.

Here is a [ZIP gallery of images](#) you can use to test your program.

4. JavaScript - Flower Garden:

Write the necessary JavaScript code `garden.js` for the functionality of the following page:

Flower Garden



of flower patches:

This page allows the user to plant a number of flower patches in their garden.

- The input box for the # of flower patches has an id of "count".
- The "Plant Garden" button has an id of "plant".
- The "Clear" button has an id of "clear".
- When the user clicks on "Plant Garden" the page should generate the # of flower patches given in the input box on the page.
- The flower patches are images; either `daffodils.jpg` or `roses.jpg`.
- Choose randomly what kind of flower patch to show.
- Clicking on the "Clear" button removes all flower patches from the garden.

5. Ajax/JavaScript - Music Search:

Write the necessary JavaScript code for the Ajax functionality of the following page.

Music Search

Search your music:

Scratching Hearts - Our Favorite Year - 3:48

Dear Hip Hop - Magnetic North - 4:04

Within The Rhythm - Magnetic North - 4:01

This page allows the user to search for songs. You are given access to a web service [music.php](#). To use the service, you may send in a query parameter named `term`. The service will then return all of the songs that contain the search term in the following XML format.

If you send the empty string "", the service will return a list of all songs.

```
<songs>
  <song genre="Pop" duration="3:48">
    <title>Dance Like Michael Jackson</title>
    <artist>The Far East Movement</artist>
    <album>Animal</album>
  </song>

  <song genre="Soundtrack" duration="3:11">
    <title>Aaj Ki Raat</title>
    <artist>A. R. Rahman</artist>
    <album>Slumdog Millionaire</album>
  </song>
</songs>
```

You will need to parse the XML to display all the songs returned to the page.

- The search button has an id of "search"
- The search textfield has an id of "query"
- Each song is displayed as a paragraph in the div with an id of "results".
- For each song returned, you must display the following separated by hyphens:
 - Title of the song
 - Artist of the song
 - Duration of the song
- You may assume that the request will be successful; you do not need to write any `onFailure` or `onException` functions.

Here is a [runnable solution](#).

6. SQL - Duplicate Cities:

Some countries have two cities with the same name. For example, there are several cities in the USA named Springfield, which is part of the reason that name was chosen for the home town of the Simpsons.

Using the `world` database, find all such cities where there are two or more cities within the same country that have the same name. Show both the country name and the city name. Eliminate duplicate results and order by the country name, breaking ties by the city name.

Recall the `world` database schema:

code	name	continent	independence_year	population	gnp	head_of_state	...
AFG	Afghanistan	Asia	1919	22720000	5976.0	Mohammad Omar	...
NLD	Netherlands	Europe	1581	15864000	371362.0	Beatrix	...
...

id	name	country_code	district	population
3793	New York	USA	New York	8008278
1	Los Angeles	USA	California	3694820
...

country_code	language	official	percentage
AFG	Pashto	T	52.4
NLD	Dutch	T	95.6
...

This is the output you should get:

```

+-----+-----+
| country | city |
+-----+-----+
| China   | Jining |
| China   | Jinzhou |
| China   | Kaiyuan |
| China   | Suzhou |
| China   | Yichun |
| Indonesia | Depok |
| Mexico  | Guadalupe |
| Mexico  | La Paz |
| Mexico  | Matamoros |
| Philippines | San Carlos |
| Philippines | San Fernando |
| Philippines | San Jose |
| Russian Federation | Zeleznogorsk |
| United States | Arlington |
| United States | Aurora |
| United States | Columbus |
| United States | Glendale |
| United States | Kansas City |
| United States | Pasadena |
| United States | Peoria |
| United States | Richmond |
| United States | Springfield |
+-----+-----+
22 rows in set (14.48 sec)

```