# University of Washington, CSE 190 M, Spring 2008
# Lab 10: Final Exam Review (Thursday, June 5, 2008)

The purpose of this lab is to practice sample final exam problems — *with pencil and paper* — in preparation for the real final exam on **Thursday, June 12, 2008**.

You won't have enough time to finish all of the exercises; finish as much as you can within the allotted time. **We suggest that you skip problems 1 and 5 for now and focus on problems 2-4.**

## 1. JavaScript/DOM/Prototype

Write JavaScript code that can be linked to an HTML page in a `script` tag so that when the page loads, every table in the page will become "zebra striped." Zebra striping in this case means that every odd row of the table will be colored to have a black background and white text. (The page's CSS file has a class named `zebrastripe` that you can attach to the odd rows if you like.) For example, the following page should have the following appearance after your code runs:

**before**

| Name | Bounces |
|---|---|
| Snuggles | 7 |
| Horseface | 13 |
| Cornelius | 3 |
| Tigger | 1000 |

| Name | Favorite Pasta |
|---|---|
| Snuggles | Fetuccini Alfredo |
| Horseface | Spaghetti Pomodoro |
| Cornelius | Chicken Penne |

**after**

| Name | Bounces |
|---|---|
| Snuggles | 7 |
| Horseface | 13 |
| Cornelius | 3 |
| Tigger | 1000 |

| Name | Favorite Pasta |
|---|---|
| Snuggles | Fetuccini Alfredo |
| Horseface | Spaghetti Pomodoro |
| Cornelius | Chicken Penne |

Note that each table has the same counting for zebra striping, i.e., the first, third, fifth, ... rows are colored. The number of rows in one table should not affect the striping for another table.

## 2. Ajax/XML

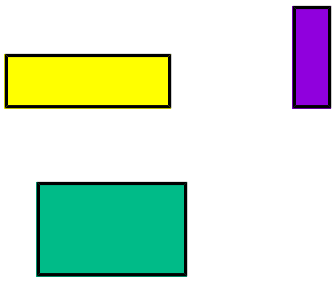
Write the complete JavaScript code to fetch XML data about rectangles and display it on the current page. The data comes from a PHP web service named `rect.php` that is located in the same directory as your code. When this service is contacted with a GET or POST request (no parameters are necessary), it outputs data about a group of rectangles to draw on the page, including the x/y position, width, height, and color of each. Your code should process the XML and display each rectangle on the page as an absolutely-positioned `div` of the appropriate position/size/color. Add the `div` to the area of the page with `id` of `rectanglearea`.

```
...
<body>
  <div id="rectanglearea"></div>
</body>
...
```

The XML data from `rect.php` will be in a format that matches the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<shapes>
  <rectangle x="10"  y="40"  width="100" height="30" color="FFFF00" /
  <rectangle x="190" y="10"  width="20"  height="60" color="9000DD" /
  <rectangle x="30"  y="120" width="90"  height="55" color="00BB88" /
</shapes>
```

For the XML data above, your code would produce the following content on the HTML page:



## 3. PHP

Write a PHP web service that would be saved as `lookup.php`. Your service will look up a name in a company's records and output a URL of information about this employee. Your script will be given a full name (first and last) via a GET request parameter named `name`. You may assume that the `name` query parameter is passed (you don't have to check whether the parameter is set).

Read data from a file named `employees.txt` and attempt to match the given name. The file has one employee's data per line, with each piece of information (full name, username, position) separated by one or more tab (`\t`) characters. The lines of the `employees.txt` file will look like this:

```
Morgan Doocy          mdoocy    Devourer of Souls!!!!
Conner Q. Reilly      conn      Director of Archives
Marla Jeffries        mjeff     Lamination Tzar
Marty Stepp           stepp     Regional Purple-Cow-Costume Chairperson
Victoria Kirst        vkirst    Chief "Yes Girl"
```

If the name is matched, output a URL in plain text representing the employee's name and position, in the following format (with any non-alphabetic characters removed from the person's position):

`http://www.awesomeco.com/`**`PositionNameWithoutSpaces`**`/`**`userName`**

For example, `lookup.php?name=Conner+Q.+Reilly` produces the following output:

```
http://www.awesomeco.com/DirectorofArchives/conn
```

If the name is not matched, output the following URL:

```
http://www.awesomeco.com/
```

## 4. SQL

Write an SQL query that will match up all actors who share the same last name and appeared in a movie together. Display the actors' first names, shared last name, and movie name. You should not match up an actor with him/herself. Order the results by last name in ABC order. The following is a subset of the results returned:

```
+-----------------+------------+-----------+---------------------+
| first_name      | first_name | last_name | name                |
+-----------------+------------+-----------+---------------------+
| Bill (I)        | Frankie J. | Allison   | Ocean's Eleven      |
| Anthony         | Frankie J. | Allison   | Ocean's Eleven      |
| Anthony         | Bill (I)   | Allison   | Ocean's Eleven      |
| J. Todd         | Sharon     | Anderson  | Fargo               |
| ...             | ...        | ...       | ...                 |
| Elenor          | Rowan      | Witt      | Matrix, The         |
| Ben (I)         | Samuel E.  | Wright    | Little Mermaid, The |
+-----------------+------------+-----------+---------------------+
134 rows in set (3.14 sec)
```

Recall that the `imdb` database contains the following tables:

**actors**

| id | first_name | last_name | gender |
|---|---|---|---|
| 433259 | William | Shatner | M |
| 797926 | Britney | Spears | F |
| ... | | | |

**movies**

| id | name | year | rating |
|---|---|---|---|
| 112290 | Fight Club | 1999 | 8.5 |
| 210511 | Memento | 2000 | 8.7 |
| ... | | | |

**roles**

| actor_id | movie_id | role |
|---|---|---|
| 433259 | 313398 | James T. Kirk |
| 797926 | 342189 | Herself |
| ... | | |

## 5. PHP+HTML+SQL

Write the PHP code to display all the results from the preceding SQL query in two ordered lists. The first list should show last names that start with A-J inclusive, and the second list should show the remaining last names that start with K-Z. Write a function named `display_last_names` that can be called to display these two lists, as shown in the PHP code below:

```
<html>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
 "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html>
    <head>
        <title>IMDB Last-name Movies</title>
    </head>
    <body>

<?php
display_last_names();
?>

    </body>
</html>
```

Your code should produce the following (abbreviated by ...) output:

```
<ol>
    <li>Allison, Anthony / Bill (I): Ocean's Eleven</li>
    <li>Allison, Anthony / Frankie J.: Ocean's Eleven</li>
    <li>Allison, Bill (I) / Frankie J.: Ocean's Eleven</li>
    <li>Anderson, George (IV) / Jo (I): JFK</li>
    ...
    <li>Jones, Annette / James Earl: Star Wars</li>
    <li>Jones, Annette / Linda (I): Star Wars</li>
    <li>Jones, James Earl / Linda (I): Star Wars</li>
</ol>
<ol>
    <li>Kennedy, Caroline / Ethel (II): JFK</li>
    <li>Kennedy, Caroline / Jacqueline (I): JFK</li>
    ...
    <li>Witt, Elenor / Rowan: Matrix, The</li>
    <li>Wright, Ben (I) / Samuel E.: Little Mermaid, The</li>
</ol>
```

You may have to slightly modify your query to give unique names to columns so that you can access these columns in your PHP code. For full credit, use embedded PHP and not `print`/`echo` statements. You do not need to check for any MySQL-related errors.