CSE 190 M, Spring 2008
Final Exam, Thursday, June 12, 2008

Name: _____

Student ID #: _____

Section and/or TA: _____

- You have **110 minutes** to complete this exam.
  You may receive a deduction if you keep working after the instructor calls for papers.
- This test is open-book/notes.
- You may not use any computing devices of any kind including calculators, cell phones, or music players.
- Please do not abbreviate any code on your exam.
- If you enter the room, you must turn in an exam before you leave.
- You must show your Student ID to a TA or instructor for your exam to be accepted.

Good luck!

Problem	Description	Earned	Max
1	JavaScript/DOM		25
2	Ajax/XML		25
3	PHP		25
4	SQL		25
X	Extra Credit		+1
TOTAL			100

1. Javascript/DOM (25 points)

Write the Javascript code to add behavior to the following HTML code. The page contains a text input box with `id` of `foodname`, and a drop-down list with an `id` of `foodgroup`. The user types a name of a food item into the `foodname` text box, such as `apple` or `Cookie`, selects a food group from the drop-down list, such as `dairy` or `fruit`, and then clicks the button with `id` of `eat`. When the `eat` button is clicked, any element on the page that matches all of the following criteria will be removed from the page:

- The element is an `img` element that has a `class` of `food`.
- The element's food group matches the group chosen in the `foodgroup` drop-down list. Food groups are represented as `class` attributes. This is in addition to the `food` class; recall that a `class` attribute can specify multiple classes separated by spaces. For example, a jug of milk would have the following element:
  `<img src="milkjug.jpg" class="`**`food dairy`**`" alt="milk" />`.
- The food item's name is the same as the text in the `foodname` box. The food's name is stored as the image's `alt` attribute. For example, if the user types `cookie`, only `img` elements with an `alt` of `cookie` will be removed. Your code should be case-insensitive; for example, `coOKIe` should also match images with an `alt` of `cookie`.

The relevant HTML code for the page is the following. Assume that Prototype is also included in the page.

```
<div>
    Name of Food: <input id="foodname" type="text" />
    Food Group:
    <select id="foodgroup">
        <option>dairy</option>
        <option>fats</option>
        <option>fruit</option>
        <option>meat</option>
        <option>veggies</option>
    </select>
    <button id="eat">Eat!</button>
</div>

<p>
    <img src="cookie.jpg" class="food fats" alt="cookie" />
    <img src="apple.jpg" class="fruit yummy food" alt="apple" />
    <img src="broc.jpg" class="food veggies" alt="broccoli" />
    <img src="tomato.jpg" class="fruit food" alt="tomato" />
    <img src="beefsteak.jpg" class="food meat" alt="steak" />
    <img src="milkjug.jpg" class="food dairy" alt="milk" />
    <img src="potatochips.jpg" class="fats food other" alt="chips" />
</p>
```
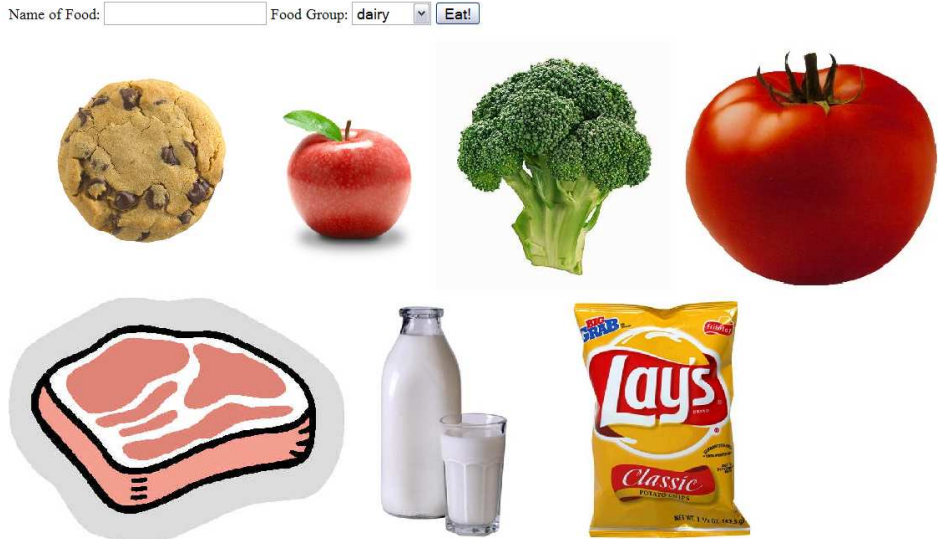

Your code should be general; it should not rely on the specific food images shown in the screenshot. You should also not rely on the exact ordering of the words in the `class` attribute. Write your answer on the next page.

1. Javascript/DOM (writing space)

2. Ajax/XML (25 points)

Write the Ajax JavaScript code to fetch and display XML data from the web service named `personality.php`, located in the same directory as your code. This code should run when the web page loads. When your code makes a GET request to this service without passing any parameters, its output is a set of XML data about users and their Keirsey personality types. Each user has a name, a 4-letter personality type, and a set of 70 answers to a personality survey. An answer can be an A, B, or a - to indicate a blank response. (This data is similar to the data used in CSE 142's personality test homework assignment.) Answers may appear in either upper or lowercase. The XML data format matches the following abbreviated example:

```
<personalities>
  <person name="The frumious bandersnatch" type="INFP">
    <answers>-BBaBAA-BBbBBABBBBA-BaBBBBBbbBBABBBBBBBABB-BBBaBBABBBBBBB-BABBBBBBBBBBBB</answers>
  </person>

  <person name="Minnie Mouse" type="ISTJ">
    <answers>BABA-AABABBBAABAABA-ABABAAAB-ABAAAAAA-AAAABAAABAAABAAAAAB-ABBAAAAAAAAA</answers>
  </person>
  ...
</personalities>
```

Your JavaScript code should fetch the list of persons and their types, and should turn each person's information into an item in an unordered list on the page with `id` of `output` that is already present on the page. Each list item should show the person's name, Keirsey type, and the count of how many total "B" (or "b") answers the person gave. For the XML data above, your code would produce the following HTML content (abbreviated with ...):

- The frumious bandersnatch: INFP (54 Bs)
- Minnie Mouse: ISTJ (18 Bs)
  ...

3. PHP (25 points)

Write a complete PHP web service `baby.php` that processes baby name data as seen in Homework 5. The service reports the best (lowest) popularity ranking that name has ever held in the Social Security data. Your service accepts a GET request parameter named `name`. Its output is a single line of plain text containing the best ranking number, or -1 if no `name` parameter is passed or if the name is not found in the file. The input file to read, `names.txt`, is in the following format, with each line containing a baby's first name followed by some number of popularity rankings. The sample data below has 11 rankings per line, but for full credit your code should work regardless of how many rankings (1 or more) are on each line.

```
Martha 31 25 24 26 30 46 93 163 209 289 382
Martin 66 79 84 94 93 78 70 108 127 142 177
Martina 631 752 712 664 720 933 636 752 680 856 918
...
```

For example, if your service were requested as `baby.php?name=Martha`, its output would be `24`, since that is Martha's best popularity ranking. If the request is `baby.php?name=Martholomew`, its output would be `-1`, since that name is not in the file. Your code should match case-insensitively; for example, the request `baby.php?name=maRTIn` should match the name `Martin` in the data file.

4. SQL (25 points)

The Springfield Elementary school board is trying to crack down on grade inflation. To do this, they are trying to figure out which teachers are giving a lot of high grades in their courses. They have asked you to write an SQL query that can be run on the simpsons database that will find the names and course names of all teachers who gave 2 or more grades of C- or better in a given course. The query should show the teacher's name and the course name. Recall that the simpsons database contains the following tables:

grades

student_id	course_id	grade
123	10001	B-
123	10002	C
456	10001	B+
888	10002	A+
888	10003	A+
404	10004	D+

students

id	name	email
123	Bart	bart@fox.com
456	Milhouse	milhouse@fox.com
888	Lisa	lisa@fox.com
404	Ralph	ralph@fox.com

courses

id	name	teacher_id
10001	Computer Science 142	1234
10002	Computer Science 143	5678
10003	Computer Science 190M	9012
10004	Informatics 100	1234

teachers

id	name
1234	Krabappel
5678	Hoover
9012	Stepp

The results of the query would be the following, because Krabappel taught 142 and gave Bart a B- and Milhouse a B+, and Hoover taught 143 and gave Bart a C and Lisa an A+:

```
+-----------+----------------------+
| Krabappel | Computer Science 142 |
| Hoover    | Computer Science 143 |
+-----------+----------------------+
```

If a teacher taught more than one course with >= 2 high grades, your query should only show that teacher's name once along with any one of the courses in which the high grades were given by that teacher. Recall that a grade above C- is a string that occurs earlier than the string 'C-' in alphabetical order.

X. Extra Credit (+1 point)

Choose a particular piece of web-related software, such as a particular web browser, operating system, or text editor. Then draw a picture of what a typical user of that software looks like. For example, you could draw a picture of a typical IE web surfer, a Mac user, a TextPad programmer, etc. Make sure to caption your drawing if necessary so that we know what software the person is using.

(Any drawing that appears to have taken more than a moment's work will get the +1 point.)