CSE 190 M, Spring 2009
Final Exam

Name: _____

Quiz Section: _____ **TA:** _____

Student ID #: _____

Rules:

- 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 use any paper resources other than practice exams.
- You may *not* use any computing devices, including calculators, cell phones, or music players.
- Unless otherwise indicated, your code will be graded on proper behavior/output, not on style.
- Please do not abbreviate code, such as writing ditto marks ("") or dot-dot-dot marks (...).
- If you enter the room, you must turn in an exam and will not be permitted to leave without doing so.
- You must show your **Student ID** to a TA or instructor for your submitted exam to be accepted.

Good luck! You can do it!

Problem	Description	Earned	Max
1	HTML / CSS Tracing		20
2	PHP		20
3	JavaScript / DOM		20
4	Ajax / XML		20
5	SQL		20
X	Extra Credit		1
TOTAL	**Total Points**		**100**

1. HTML / CSS Tracing

Draw a picture of how the following HTML/CSS code will look when the browser renders it on-screen. Assume that the HTML is wrapped in a valid full page with a `head` and `body`. Indicate a non-white background by shading lightly or by drawing diagonal lines like this. It is possible that some CSS rules shown will not apply to any elements.

HTML:

```
<p class="a b">If you only knew
                  the power of the Dark Side.</p>

<ul class="a">
    <li>Luke Skywalker</li>  <li>Princess Leia</li>
    <li>Han Solo</li>  <li>Chewbacca</li>
    <li>C3-PO</li>  <li>R2-D2</li>
</ul>

<p class="b c">Emperor Palpatine (to Luke Skywalker): I'm afraid the
<span>deflector shield</span> will be quite operational when your friends arrive.</p>

<p id="b" class="d">Luke: That's not true.
That's impossible!</p>

<div class="b">
<h1 id="d" class="c">May the Force       be with you</h1> </div>
```

CSS:

```
.a {
    background-color: yellow;
    float: right;
    width: 10em;
}

.b .c {
    text-decoration: underline;
    background-color: yellow;
}

.a, #b {
    border: 2px solid black;
}

#c, p span {
    border: 2px dotted black;
    padding-left: 2em;
}

.c #b, .b > .c {
    clear: both;
}

#d {
    text-align: center;
}
```

1. **HTML / CSS Tracing (writing space)**

2. PHP

Write a complete PHP web service that could be saved in a file `ascii.php`, that outputs a single frame from a student's `asciimation.txt` file from the ASCIImation assignment. Each student's ASCII data is stored on the web server with your PHP file, in a students directory within subdirectories that have the same names as the students. For example, student `jsmith`'s ASCII data is stored in `students/jsmith/asciimation.txt`. Suppose that the contents of that file are those shown below.

The service is sent a request parameter named `student` that corresponds to a student whose ASCIImation data should be retrieved. The service also accepts an optional second parameter named `frame` specifying which frame of the animation to output, 0-based. If this parameter is not passed, frame 0 (the first frame) is used. Recall that frames are separated by a line of `=====` . So for example, if your web service is contacted using the URL of `ascii.php?student=jsmith&frame=2` , the following should be the output:

students/jsmith/asciimation.txt	ascii.php?student=jmith&frame=2 , output

<pre> o
  O
 o/|\o
  / \
=====
  o
 o_O
  |o
  / \
=====
 o o
  \O
  o\
  / \
=====
  o
  _O_o
  o|
  / \</pre>
	<pre> o o
	  \O
	  o\
	  / \</pre>

Your output should use the `text/plain` content type. If the student parameter is not passed or refers to a non-existing file/directory, the service produces no output or error. If the `frame` parameter is passed with a negative value, or a value greater than the number of frames in the student's file, no output or error is produced.

Write your answer on the next page.

2. PHP (writing space)

3. JavaScript / DOM

Write the Javascript code to add behavior to the following HTML code. The page is similar to the number guessing game assigned in CSE 142. When the page loads, your code should randomly choose a number from 1-100 inclusive. (Recall that the Math.random function returns a random real number between 0 and 1.)

When the user types a number into the number text field and then clicks the makeguess button, the game will compare the user's guess to your randomly chosen number, and report whether it was "Too high!", "Too low!", or correct. The information will be shown as text in the result span. If the guess is correct, you should show a message such as, "You got it right in 6 tries!" (You can still say "tries" even if it takes 1 guess.) Once the user guesses correctly, disable the button so that no more guesses can be made.

The game will also show a history of all guesses made as a bulleted list. Each guess's number is added as a bullet to the end of the list. If the guess was too low, this bullet should use the low CSS class, which displays it in blue italic. If it's too high, the bullet should use the high CSS class, which is red and bold. If it is correct, it should be displayed without any class.

You may assume that the text the user types in the field can be interpreted as an integer. You don't need to worry about the case where the user makes the same guess twice. You may assume that Prototype is also included in the page.

The relevant HTML/CSS code for the page is the following.

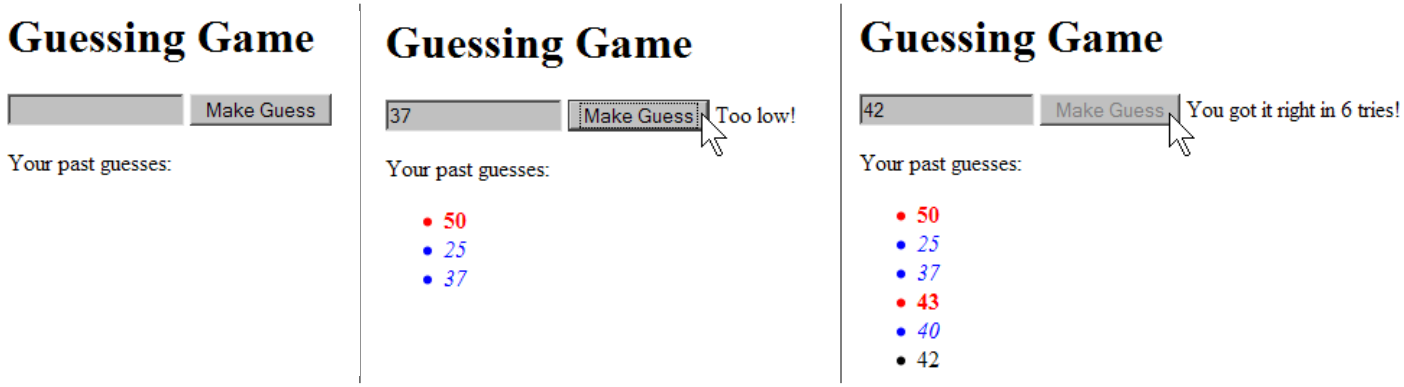
```
<h1>Guessing Game</h1>
<div>
    <input id="number" type="text" size="16" />
    <button id="makeguess">Make Guess</button>
    <span id="result"></span>
</div>

<p>Your past guesses:</p>
<ul id="guesses"></ul>
```

```
li.low {
    color: blue;
    font-style: italic;
}

li.high {
    color: red;
    font-weight: bold;
}
```

The following screenshots show the initial state and state after several guesses have been made.



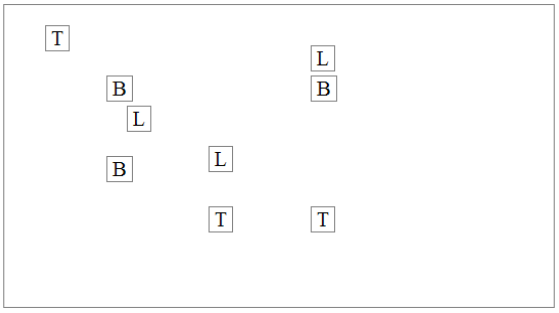
Write your answer on the next page.

3. **JavaScript / DOM (writing space)**

4. Ajax / XML

In this problem we have a web page that displays letters that represent "Critter" animals in a 2D simulation. Each critter is represented with a `span` that has been absolute-positioned to give it an x/y location on the page. The relevant page code and its appearance are shown below:

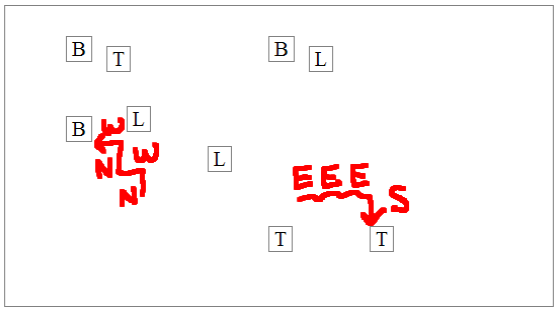
```
<h1>Critters</h1>
<div id="critterarea">
    <span class="critter Bear">B</span>
    <span class="critter Bear">B</span>
    <span class="critter Bear">B</span>
    <span class="critter Lion">L</span>
    <span class="critter Lion">L</span>
    <span class="critter Lion">L</span>
    <span class="critter Tiger">T</span>
    <span class="critter Tiger">T</span>
    <span class="critter Tiger">T</span>
</div>
```



Suppose that there is a web service named `critter.php`, located on your web server in the same directory as your code. This service outputs XML data describing sequences of moves to make on various critters on the page. In this problem you should write Ajax JavaScript code to contact the web service (using a GET request), examine its XML data, and move the corresponding critters appropriately. This code should run when the web page loads.

The XML data returned by the web service consists of a series of one or more `move` elements inside an overall `movelist` element. Each `move` element contains an `animal` attribute describing the class of animals (such as bears or lions) that should be moved and a letter indicating the direction to move. The format matches the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<movelist>
    <move animal="Bear">N</move>
    <move animal="Bear">W</move>
    <move animal="Tiger">E</move>
    <move animal="Bear">N</move>
    <move animal="Bear">W</move>
    <move animal="Tiger">E</move>
    <move animal="Tiger">E</move>
    <move animal="Tiger">S</move>
</movelist>
```



The letter in the `move` data indicates a move of 20 pixels in a given direction. N means north (up on the page), S is south (down), E is east (right), and W is west (left). For example, when processing the `move` elements above, your code should cause all `Bear` critters on the page to move 20px up (N), then 20px left (W), then 20px up (N), then 20px left (W). The expected page appearance after processing the given XML data is shown at right.

Your code doesn't need to worry about animals colliding with each other, walking off the screen, or making the movement animate. You may assume that Prototype is already linked to the page.

Write your answer on the next page.

4. Ajax / XML (writing space)

5. SQL

Write an SQL query that searches the `imdb` or `imdb_small` database for all actors who have been in 3 or more total movies, and have been in at least one action film (a movie with the "Action" genre). Your result set should include each actor's first and last name, sorted by last name and then by first name. Each actor should be listed only once.

Recall the `imdb` database tables:

actors

id	first_name	last_name	gender
433259	William	Shatner	M
797926	Britney	Spears	F
831289	Sigourney	Weaver	F
...			

movies

id	name	year	rank
112290	Fight Club	1999	8.5
209658	Meet the Parents	2000	7
210511	Memento	2000	8.7
...			

roles

actor_id	movie_id	role
433259	313398	Capt. James T. Kirk
433259	407323	Sgt. T.J. Hooker
797926	342189	Herself
...		

directors

id	first_name	last_name
24758	David	Fincher
66965	Jay	Roach
72723	William	Shatner
...		

movies_directors

director_id	movie_id
24758	112290
66965	209658
72723	313398
...	

movies_genres

movie_id	genre
209658	Comedy
313398	Action
313398	Sci-Fi
...	

When run on the `imdb_small` database, your query would produce the following results:

```
+-------------+-----------+
| first_name  | last_name |
+-------------+-----------+
| Steve       | Buscemi   |
| Michael (I) | Madsen    |
| Bill        | Paxton    |
| Stevo       | Polyi     |
| Uma         | Thurman   |
+-------------+-----------+
```

Hint: You may be tempted to write a very long query, but you only really care about the genre of one of the actor's films. The others can be anything as long as there are 3 distinct films the actor has been in.

If you join too many tables together that are not needed for the query, you will not receive full credit.
You should solve this problem using only the SQL syntax shown in class and the textbook.

X. Extra Credit

Draw a picture of your TA surfing one of his/her favorite web sites.

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