Melissa's CSE 154 Midterm Exam Strategies – Autumn 2017 o(^▽^)o

General Test-taking tips

- Breathe. You're here to learn. You've successfully completed the homework assignments, section problems, and labs, and it's just a matter of applying what you've learned to different problems!
- Practice early so you don't cram the day before! A clear head is essential to doing well on a CSE exam, as there will be a lot of logic involved. With that, it's perhaps most important to get at least 8 hours of sleep the night before to be able to have a sharp mind when the time comes to solve new problems with familiar topics.
- I have included selected problems that I recommend doing for each category, all of which are of exam-level difficulty and possibly offer some good twists to be prepared for.
- Read the problem description at least once before attempting. Underline any small details (case insensitivity, error handling, query parameters, etc.)
- Then read the problem once more *carefully* after you've finished your solution. It's easy to forget the small details, and it's worth the 1-2 minutes to double-check. Seriously, this is **strongly** recommended.
- When studying, **study what you don't want to study.** It's a common habit of students to avoid what they're dreading, but this is unfortunately a recipe for not being prepared when the time comes to solve the type of problem on the exam. For example, if you still don't understand cookies or sessions, watch the lecture video on these again.
- Give yourself space between statements to account for any last-minute fixes/special cases

HTML Validation

- HTML problems on midterms typically involve finding validation errors in provided HTML
- Validation errors may include missing opening/closing tags, having an inline element outside of a block element, using a tag instead of an HTML attribute (e.g., <href> instead of href in <a>), misuse of <link> or <script> (there are various differences between the syntax of these two <head> tags), etc.
- Tips for studying:
  - Review the HTML portion of the CSE 154 style guide
  - Go through the recommended problems from past exams
- Recommended problems
  - Problem 1 16au
  - Problem 1 16au extra practice exam
  - Problem 1 17sp

CSS Writing/Query Selector Mystery

- CSS problems have historically been in a couple of different formats, including:
  - Writing CSS to match a given screenshot with provided HTML
    - Problem 1 11sp (more "web-pagey")
    - Problem 2 13sp (more "div-and-float-puzzly")
    - Problem 2.c. 16au (example of a smaller CSS-writing problem from more recent quarters)
    - Problem 3 16au extra practice exam
    - abbysStyleGuide (CodeStepByStep)
    - Problems 1, 2, and 3 of extra questions

- Given HTML, write the CSS query selectors to select different tags for the HTML
    - Final 17sp 2.b
- Given HTML and CSS query selectors, writing the different tags that each selectors matches
    - Final 17sp 2.a.
    - Final 16au 2.a.
    - Problem 2 16au extra practice exam
    - selectorMystery1 (CodeStepByStep)
    - lotrSelectors (CodeStepByStep)

- More recent quarters have done the last two types of CSS problems to avoid the time-crunch that many students found with writing CSS from a full spec, but you should make to be comfortable approach any of these types of problems
- When writing CSS, make sure you add **all** of the required styles. It's recommended to first follow the criteria listed explicitly when adding each CSS property, and then check what you have so far with the provided image. There are some properties that may not be listed explicitly on the criteria, and there is some criteria that you can't know just by looking at the picture (e.g., a background-color value or font-size)
- For selector mystery problems, make sure to review the different types of query selectors, such as:
    - p li
    - p > li
    - p > li > li
- Common errors:
    - Forgetting to handle small details (italic or bold text on one span element, text alignment, overflow, clear, and float properties, etc.)
    - Improper use of floating (remember that you can't use something like float: center to center a div)
    - Treating a block element as an inline element and vice versa. These two types behave differently for different CSS properties.
    - Not taking inherited relationships between elements into consideration. For example, if a div inside a div has width set to 50%, it's 50% of the width relative to the parent div.
    - Centering a block element as you would text
        - For block elements, use margin: auto auto;
        - For text elements, use text-align: center;
    - What are *your* most common errors for these problems?
- Tips for studying this type of problem:
    - Take out a piece of paper. Take 5 minutes and 1.) write out all of the **CSS properties** (e.g., “float”, “border”, “font-family”, etc.) you can recall, 2.) the different **values** you could give each, and 3.) the **different HTML tags** you might need to call the property on (e.g., <div>, <p>, <h1>, <img>, etc.). At the end of the 5 minutes, review any properties you may have felt less confident about, and make sure you can know how to use them for different elements. There's a nifty list of CSS properties here to check.
    - Use codepen.io to play with different CSS properties
    - Understand the differences between block vs. inline elements. Refer to this handy cheatsheet for different tags and their types.
        - Understand the differences between ID's and classes

- Understand the box model and how to use it (e.g., padding vs. margin)
  - Review problem: boxes
- Study floating vs. positioning vs. flexbox
  - Review problem: flexBoxes
- Visit your favorite web page, copy/paste the HTML, and try to duplicate the appearance using CSS. Depending on the complexity of the page, this may be more advanced than you're used to, but is probably the best practice you could get. Avoid HTML with <table> (the developers hacked the layout).

JavaScript/DOM/Events

- Given a description of a page's event-driven behavior, write the JavaScript code to implement that behavior
- May involve:
  - Event listeners on input elements
  - DOM manipulation
  - Timers/animation
- Common errors:
  - Not correctly accessing values of different input elements (how do drop-down vs. radio button vs. checkboxes vs. text input values differ?)
  - Not handling "case-sensitivity" for text if relevant/specified
  - Forgetting to use parseInt on strings with integer values
- Tips for studying this type of problem:
  - Review events and timers
  - Helpful JavaScript overview/glossary
  - Know how to navigate parent/children/sibling DOM elements
  - Know how and when to append new elements to the page using JavaScript
  - Remember to fetch elements on the page using document.getElementById (or ID) and querySelector (or QS)
- Recommended problems:
  - Problem 3 17sp (DOM manipulation, setting position/style to added divs)
  - Problem 2 12sp (input text processing and DOM manipulation)
  - Problem 4 13sp (input text processing and DOM manipulation)
  - Problem 2 14sp (to-do list processing, adding and removing elements)
  - Section 4 17au (animations, DOM manipulation, event handlers)
  - Various JavaScript practice problems (CodeStepByStep)

Using a PHP Web Service with JavaScript and AJAX

- This type of problem asks to use a given PHP web service (e.g., "quotes.php") by
  - Getting some value(s) or onclick actions from user input on an HTML page
  - Using these values to request data from the web service using AJAX, and outputting the results as HTML **using JS** (e.g., the DOM methods)
  - You will only need to use GET requests, so ignore any questions with POST requests
  - Remember how to use the fetch syntax we've introduced in lecture this quarter!

- Common errors:
  - Not calling your AJAX request method if needed when the page loads vs. when something is submitted/clicked
  - Forgetting to pass parameters to the URL when building an AJAX request (if applicable)
  - Not properly processing result text (methods differ between plain text, HTML, and JSON)
- Tips for reviewing this type of problem:
  - Know how to and practice:
    - Extracting data from response (HTML, text, or JSON)
    - Creating and appending HTML elements into page
- Example problems
  - Problem 4 16au (using JSON)
  - Problem 6 16au extra practice exam (using JSON)
  - Problem 2 of 12au (using JSON)
  - Problem 3 of 14sp (using JSON)
  - Section 5 17au (text, JSON, and HTML AJAX exercises)

Short Answer Questions

- Some exams have a "short answer" section of the exam with 5-10 short answer questions. These questions range in topic but aim at covering each week of lecture topics.
- Tips for studying:
  - Go over lecture slides! These are conceptual, so the lecture slides are the best way to cover the higher-level concepts.
- Example problems
  - Problem 8 17sp (only focus on those that are HTML/CSS/JS related and ignore those that are server/PHP/SQL/Regular expression related)
  - Problem 9 16au extra practice exam (again, only focus on relevant material for midterm)
  - Various "web programming basics" practice problems (CodeStepByStep)