Test Review for Part Two

1. Warning: Not Complete

This review packet includes a lot of material, but it is not complete. Use it as a guide, but use your class notes, lab and homework projects to practice. Do not just read things; you will need to be able to do things. If you notice a certain amount of repetition, that is the result of cutting and pasting old review sheets.

2. Do Exercises, Experiment with Code

Using computers is an activity, not a spectator sport. The only way you gain comfort and fluency with a language, spoken or computer-oriented, is through speaking and using the language. Using a computer language means writing scripts or typing commands that get the computer to do things you want them to.

The class handouts include sample programs we examined and ran in class. You can copy these programs from the Classes links on the course web site. The best way to understand these programs and the ideas they demonstrate is to type them (or copy them), run them, then tinker with them. Change some of the code to see what happens. Add some new features or modify existing ones. When something surprising or confusing happens, note your questions and bring them to office hours.

On the other hand, part of the course looks at ideas, terms, vocabulary, and skills. You have to know terms and facts to use the language meaningfully.

3. Main Topics

Since the last exam we looked at two main areas: digital representation of information and programming in Javascript. The main aspects are:

   a. Representing information with lists of numbers: text, images, sounds, movies
   b. Representing numbers in binary
   c. Representing binary in physical form: holes, relays, tubes, magnets, transistors
   d. Elements and Attributes, Functions and Events

4. Digital Representation of Text, Images, Sounds

The course began by looking at machines to play music and animate images. Music was recorded in holes in cards, in bumps on metal disks, in gears in machines. Images were animated by gears driven by some sort of coding. Those ideas led to the digitization of information.

Chapters 6 and 12 explain the history of computing and the principles of representing information in digital form. Programmable machines date back to some music boxes we saw in the first class. Some books point on the Jacquard loom as an early example of a programmable machine.

But the main history of machines to process digital representation of information begins in the late part of the 19th Century. This history is divided into various generations: (0) mechanical, (1) vacuum tubes, (2) transistors, (3) integrated circuits, (4) very large scale integrated circuits, and (5) parallel and network processing. Moore’s law has accurately predicted that the number of transistors that can be fit on a chip doubles every 12-18 months.

The personal computer revolution began in 1975 when the Intel 8080 chip a fairly powerful processor affordable to ordinary people (about $500 for a computer kit).

In Chapter 12 you read about digital representation of information. Computers store, process, and transmit information as lists of numbers. The ASCII code uses 7-bit numbers to stand for characters. Therefore a text document is just a list of numbers. Music can be stored in midi format as a list of numbers standing for notes or in sampled format as a list of numbers standing for wave amplitudes. Assigning numbers to colors (or grayscales) allows us to represent images as a list of numbers, each standing for a pixel.

The numbers can be represented in binary, a numbering system that uses 1’s and 0’s to stand for the
quantities of 1’s, 2’s, 4’s, 8’s, 16’s etc. Finally, 1’s and 0’s can be represented by physical things. Holes in cards, magnetized bits of iron, open or closed switches, pits and flat surfaces on a CD, pulses of light, pulses of electricity, even just marks on a paper tape can represent strings of 1’s and 0’s, that is represent lists of number, that is represent all sorts of information.

Terms and Ideas: Be able to explain/define these terms

- ASCII
- pixel
- 2-bit color
- 3-bit color
- 8-bit grayscale
- Red/Green/Blue 24-bit color
- amplitude
- sample rate
- 44,000
- midi
- resolution
- accuracy
- compression
- binary
- decimal
- core memory
- An Wang
- punch cards

Skills: Be able to do each of these

- translate between binary and decimal
- translate an image into binary and back
- translate a wave into binary and back
- translate text into binary and back (with an ASCII table)

5. Programming Web Pages

The second main topic of this unit was the start of programming in Javascript to make web pages more interactive. Specific topics include:

1. Elements and Attributes
   a. All elements on a page have attributes
   b. These attributes can be read and modified by Javascript
   c. Javascript code consists of functions in <HEAD> of page
   d. Events on the page invoke functions

2. Functions
   a. Know how to read functions
   b. Know how to write functions
   c. Predict what functions will do or return for given args
   d. Functions that operate on elements
   e. Functions that take arguments
   f. Returning a value
   g. Using the return value of a function

3. Function internals
   a. variables
   b. Assignment operation: =
c. if..then..else
d. arithmetic operations
e. string concatenation
f. boolean expressions
g. boolean operations: &&, ||, !
i. comparison operations: <, >, ==, <=, >=, !

4. Basic Techniques
   a. counter
   b. total
   c. swapping values

5. Interactive Web Pages
   a. Data Input/Output using forms in tables
   b. Data Input/Output using popups: alert, confirm, prompt

Exercise

Write the HTML code to produce a table that looks like:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>b</td>
</tr>
<tr>
<td>3</td>
<td>c</td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

6. Dynamic Web Pages: Elements, Attributes, Events, Functions

We began by seeing how elements on a webpage have properties and we saw that Javascript functions can examine and modify those properties. We also used the term attributes for properties.

6.1. Ideas/Terms/Facts

Make sure you can explain and/or give examples of each of these terms/ideas:

- element
- attributes/properties
- boolean expression
- events
- functions
- onClick
- onMouseOver
- `<script type='text/javascript'>`

What are the attributes of a image tag? What does each represent?
Which ones can you change?
What are the attributes of a button? What does each represent?
Which ones can you change?

6.2. Skills

The first set of skills we studied was how to add functions to a page, how to make those functions modify attributes of elements, and how to attach functions to specific events.

- a. Write a javascript function
- b. Include javascript functions in the head of a page
- c. Call a javascript function when an event occurs
- d. Modify the background color of the page
- e. Swap the pictures between two image tags
f. Use variables in a JavaScript function

g. Use alert() to tell a user something

h. Use prompt() to get an answer from a user.

6.3. Sample Problems

Add JavaScript and/or buttons and text areas to produce the following:

1. Add two buttons to page, one marked "blue", one marked "red". When each is clicked, the page background changes to the named color.

2. When the page loads, a pop-up window appears asking the user for a color. The background color of the page is set to that color.

3. Write a page that has three images each 120 pixels wide. Add code to the page so when the mouse is over an image the width of the image changes to 240 pixels then reverts to 120 when the mouse is no longer over the image.

4. Make a copy of the page you did for #3. Change the code so that when you click on an image it expands to a width of 240 and when you click on the image again, the width goes back to 120.

7. JavaScript Functions

The function is to JavaScript what a Script is to Scratch. A function is a sequence of statements that the browser executes from top to bottom. A function has a name and can accept arguments. A function can call other functions. A function can return a value to its caller.

There are two main types of functions: functions that act on parts of the page, and functions that process values passed to them as arguments.

An example of a function that acts on parts of the page is:

```javascript
// function that reads a string from a box and sets the bgColor
function new_bg() {
    var new_color;
    new_color = document.getElementById('colorname').value;
    document.body.style.backgroundColor = new_color;
    alert("Color changed to " + new_color + "!");
}
```

An example of a function that processes values passed to it is:

```javascript
// function to compute price of coffee and doughnuts
function food_cost(num_coffees, num_doughnuts) {
    var total;
    total = (num_coffees * 1.20) + (num_doughnuts * 0.89);
    return total;
}
```

Notice the difference between these two functions. The second function is more like a function in math: it takes some input values, does some operations on those values, then sends out a result value. On the other hand, the first function is not like a math function. It does not take input values nor does it output a result.

This distinction between functions that act on elements and functions that process data is not sharp. There are many functions that fall in between these extremes. For example:

```javascript
// A function to set a value but it takes an argument and returns a result
function set_color_if_ok(newcolor) {
    var result;

    if (confirm("Set bg to " + newcolor + "?") == true) {
        document.body.style.backgroundColor = newcolor;
        result = true;
    }
}
```


```javascript
else {
    alert("ok, no change then.");
    result = false;
}
return result;
```

7.1. Ideas/Terms/Facts

Make sure you can explain and/or give examples of each of these terms/ideas:

- local variables
- arguments
- return
- return value
- statement
- boolean expression
- numeric expression
- boolean operations (&&, ||, !)
- numeric operations (+, -, *, /)
- conditional execution (if..then..else)
- variables: set, use
- variables: global/local, public/private
- variables for counting
- variables for communication
- variables for running total (accumulator)
- recursive function

7.2. Skills

Writing, reading, using functions are the basic skills of using most modern programming languages. The following skills are essential to using Javascript effectively:

- Define a function
- Calling a function
- Passing values to a function
- Using the return value from a function
- Setting the value of a variable
- Incrementing/Decrementing a variable
- Changing a variable by a fixed amount
- Using a variable to count events
- Using a variable to add up a sequence of values
- Using a conditional expression in an "if..then..else" statement

7.3. Questions

At the end of this packet are several pages of function practice. These are functions you can trace through to predict what they will do. The code for all those samples is in /comp/10IDI/files/mt2-files if you would like to copy them to your account and try them.

In addition to those sample problems, here are some more function exercises:

1. Consider a function with two variables, x and y. When the function is called, x holds the value 5, and y holds the value 7. For each of these expressions, state the type of the expression (boolean or numeric), and state the value of the expression:
<table>
<thead>
<tr>
<th>expression</th>
<th>type</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 + 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x &lt; y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x &lt; 10 &amp;&amp; y &gt; 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(x != 2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x + 2 - y * 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-(x + 2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x = 2</td>
<td></td>
<td>x = 5</td>
</tr>
<tr>
<td>x = 2 &amp;&amp; x = 5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Write a web page with a button that says "click me" and a function so that a pop-up appears saying "Enough!" after the user clicks the button five times.

3. a. Write a webpage that adds three numbers using prompt and alert. The program asks the user for three numbers, one by one then displays the sum.
   b. Modify your answer to a so it uses text input boxes instead of prompt and alert.

4. Given that the variable x stores the value 10 and the variable y stores the value -3, what is the value of each of these expressions:
   a. 2 < 3
   b. x < y
   c. x = 10 && y = 0
   d. x = 10 || y = 0
   e. (x + y) < (2 * x)

5. At the end of this function, what does the alert show?
   ```javascript
   function mystery()
   {
   var x, y;
   x = 0;
   y = 1;
   while( x <= 20 ){
   x = x + y;
   y = y + x;
   }
   alert("y is "+y);
   }
   ```

6. In this function, how many times does the "Hello" message pop up?
   ```javascript
   function mystery2()
   {
   var x;
   x = 1;
   while( x <= 10 ){
   if ( ( x > 3 && x < 6 ) || x == 1 ){
   alert("Hello");
   }
   x = x + 2
   }
   ```

8. User Interaction: PopUps vs Tables
   Using Javascript, we can make a webpage an interactive document. The page can ask the user questions, perform computations, make decisions, and display results. There are two main ways a webpage can talk with a user: pop-ups or forms. Forms are easier to read if they are embedded in tables.

8.1. Ideas/Terms/Facts
   Make sure you can explain and/or give examples of each of these terms/ideas:
   ```javascript
   alert()
   ```
confirm()
prompt()
true
false
null
parseFloat
parseInt
isNaN()
document.getElementById('elementid').value
<input type='text' id='n' value='v' size='3' />

8.2. Skills
Creating interactive webpages requires, among others, the following skills:
   a. Using alert, confirm, prompt
   b. Creating forms
   c. Creating input elements inside of forms
   d. Reading the value from a text box
   e. Writing a value into a text box
   f. Converting the text in a text box into a number
   g. Passing values to functions
   h. Receiving return values from functions

8.3. Questions
These questions may or may not appear on a test. Writing out answers to these questions is good preparation for the test.

1. Write a webpage that, using prompt and alert, asks the user for three numbers then displays the average of those three numbers.

2. Write a webpage that, using text boxes and a button, allows the user to type in three numbers, then displays the average of those three numbers in a fourth text box.

3. Write a webpage that assists visitors to New England in planning what to wear. The page should ask, using prompt, the user for in what season he/she is planning to visit. If the user enters "winter", "spring", "summer", or "fall", the program uses alert to display a short message (like "heavy coat and gloves"). If the user enters any other word, the program displays a message saying that is not a season it knows about.

4. Write a version of your answer to #3 that uses text boxes and a button.

5. Write a webpage with an order form for a doughnut shop. The form should look like:

<table>
<thead>
<tr>
<th>Doughnut Shop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
</tr>
<tr>
<td>Doughnuts</td>
</tr>
<tr>
<td>Coffees</td>
</tr>
<tr>
<td>Tax</td>
</tr>
<tr>
<td>[Total]</td>
</tr>
</tbody>
</table>

   The user fills in the number of doughnuts and coffees, then clicks the Total button. When the button is clicked, the program fills in the cost of each item, computes the tax, then computes the total cost, filling in the boxes.