CS 416
Web Programming

Strings, arrays & Forms

Dr. Williams
Central Connecticut State University

Review from last class

- Practice: functions and conditional execution
- Given the following HTML write a function that reads the value in the text box. It then sums the numbers from the number entered until 0 and writes that value in the text box.

```
<html>
<head>
<title>Review of lecture 4</title>
</head>
<body>
  <input type="textbox" />
  <input type="button"  value="Sum" />
</body>
</html>
```

Agenda

- Strings
- Javascript arrays
- HTML form elements
  - Password
  - Radio buttons
  - Check boxes
  - Drop down lists
  - Fieldsets/Legend
  - Submit button
  - Forms and Javascript

String functions

- Similar to Java String objects, Javascript strings have similar methods
- With Javascript strings first character is at position 0
- **charAt(*pos*)** – returns character at position *pos*, so “Hi there”.charAt(0) would return “H”
- **length** – is an attribute that returns the length of the string

Other string functions

- **substring(*x*,*y*)** returns the portion of the string starting at position *x* (inclusive) and ending at position *y* (exclusive).

"Hi there".substring(1,4) returns "i t"

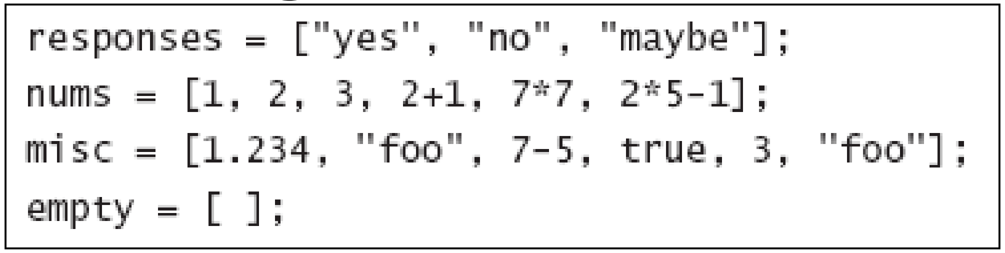
- **toUpperCase()**
- **toLowerCase()**
- **search("*abc*")** returns the index of the first character matching the string, or "-1" if not found

Differences strings and arrays

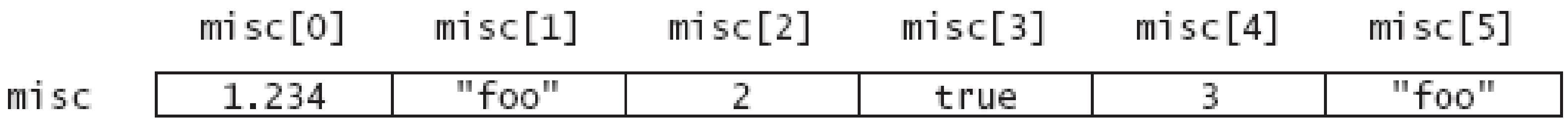
- Like strings, JavaScript *arrays* are objects that encapsulate multiple values and their associated properties and methods
- Unlike strings (which store only text characters), the items in an array can be of any data type
- Unlike most programming languages Javascript arrays can be a mix of types

Accessing items in an array

- An array stores a series of values, its associated memory cell can be envisioned as divided into components, each containing an individual value



```
responses = ["yes", "no", "maybe"];
nums = [1, 2, 3, 2+1, 7*7, 2*5-1];
misc = [1.234, "foo", 7-5, true, 3, "foo"];
empty = [ ];
```



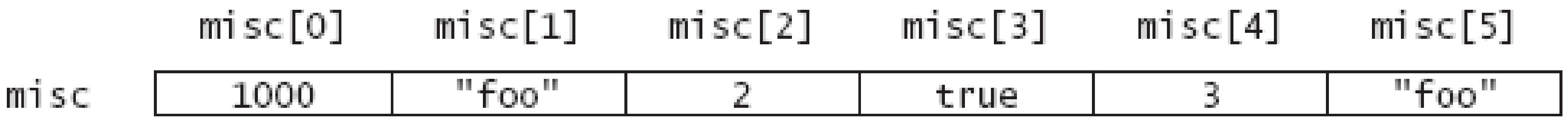
array name and then the position in square brackets ex. misc[1] to access “foo”

Assigning items in an array

- array items can be assigned values just like any variable
  - suppose the array `misc` has been assigned to store the following

```
misc = [1.234, "foo", 7-5, true, 3, "foo"];
```

  - the assignment `misc[0] = 1000` would store the value 1000 as the first item in the array, overwriting the value that was previously there



Assigning items cont.

	misc[0]	misc[1]	misc[2]	misc[3]	misc[4]	misc[5]
misc	1000	"foo"	2	true	3	"foo"

if the index in an assignment statement is beyond the array's current length, the array will automatically expand to accommodate the new item

- the assignment misc[8] = "oops" would store "oops" at index 8

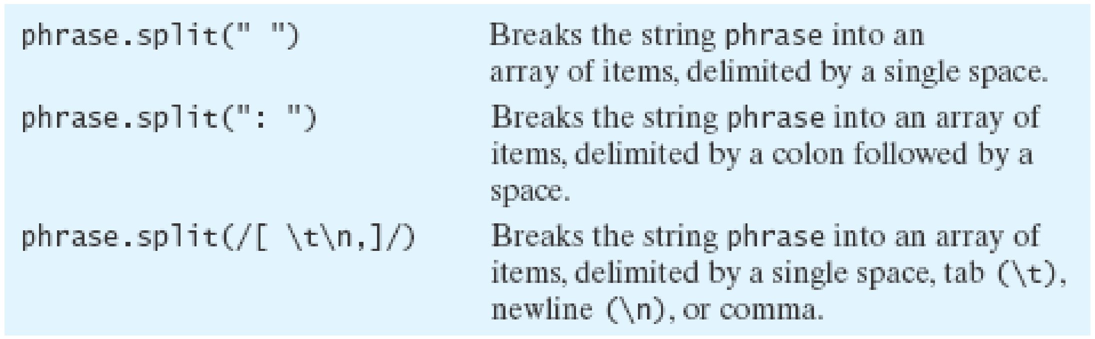
	misc[0]	misc[1]	misc[2]	misc[3]	misc[4]	misc[5]	misc[6]	misc[7]	misc[8]
misc	1000	"foo"	2	true	3	"foo"	undefined	undefined	"oops"

String `split` method

- JavaScript strings provide a method, `split`, for easily accessing the components of a string
  - the only input required by the `split` method is a character (or sequence of characters) that serves as a delimiter for breaking apart the string
  - the `split` method separates the string into component substrings at each occurrence of the delimiter, and returns an array consisting of those substrings

```
user = "Grace Murray Hopper";
arr1 = user.split(" ");       // ASSIGNS arr1 to be the array
                              //   ["Grace", "Murray", "Hopper"]
```

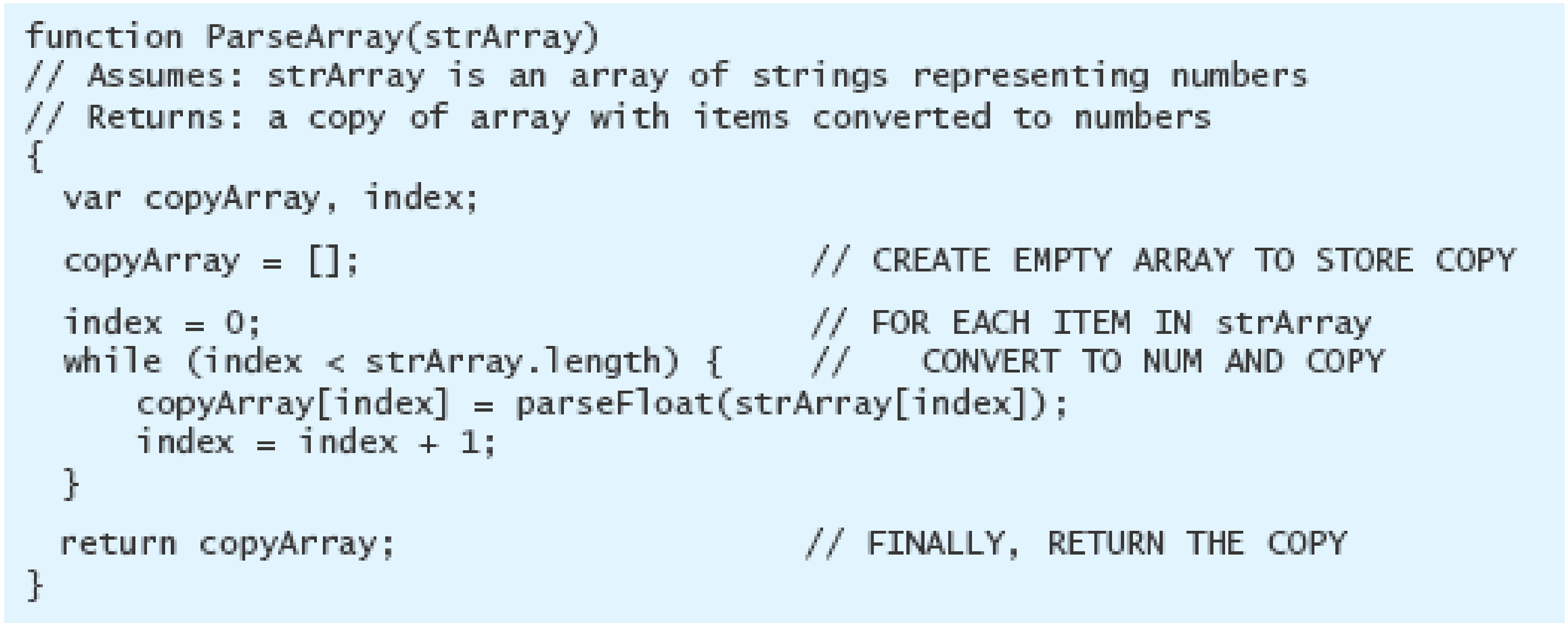
- as was the case with strings, `/[ … ]/` can be used to specify groups of characters



`phrase.split(" ")`	Breaks the string phrase into an array of items, delimited by a single space.
`phrase.split(": ")`	Breaks the string phrase into an array of items, delimited by a colon followed by a space.
`phrase.split(/[ \t\n,]/)`	Breaks the string phrase into an array of items, delimited by a single space, tab (\t), newline (\n), or comma.

Number arrays

- note: even if the input string contains numerical digits, `split` returns each array item as a string (similar to `prompt` and text box/area access)
  - we can write a function to traverse the array and convert each item to a number



```
function ParseArray(strArray)
// Assumes: strArray is an array of strings representing numbers
// Returns: a copy of array with items converted to numbers
{
  var copyArray, index;

  copyArray = [];                     // CREATE EMPTY ARRAY TO STORE COPY

  index = 0;                          // FOR EACH ITEM IN strArray
  while (index < strArray.length) {   //   CONVERT TO NUM AND COPY
      copyArray[index] = parseFloat(strArray[index]);
      index = index + 1;
  }

  return copyArray;                   // FINALLY, RETURN THE COPY
}
```

Calculating with arrays

- Say the user entered a string containing numbers separated by spaces or new line and we want to calculate the sum of the numbers

```
function getSum(){
  str = document.getElementById(“numbers”);
  strArray = str.split(/[ \n]/);
  numArray = ParseArray(strArray);
  sum = 0;
  index = 0;
  while (index < numArray.length){
    sum = sum +numArray[index];
    index = index + 1;
  }
  document.getElementById(“sum”).value = sum;
}
```

Practice – reverse

- Write a function that when given a string with words separated by commas, it outputs the words in reverse order (ie. Last word first) separated by spaces.

HTML Forms

- HTML form elements
  - Password
  - Radio buttons
  - Check boxes
  - Drop down lists
  - Fieldsets/Legend
  - Submit button
  - Forms and Javascript

Purpose of Forms

- Forms are used to pass a group of data to the server
- A form can contain input elements like
  - text fields(textbox and textarea)
  - checkboxes
  - radio-buttons
  - select lists
  - submit buttons

Form input

- The most important form element is the input element.
- The input element is used to specify various ways to collect user information
- An input element can vary in many ways, depending on the type attribute.

```
<form>
  input elements
</form>
```

Form input cont.

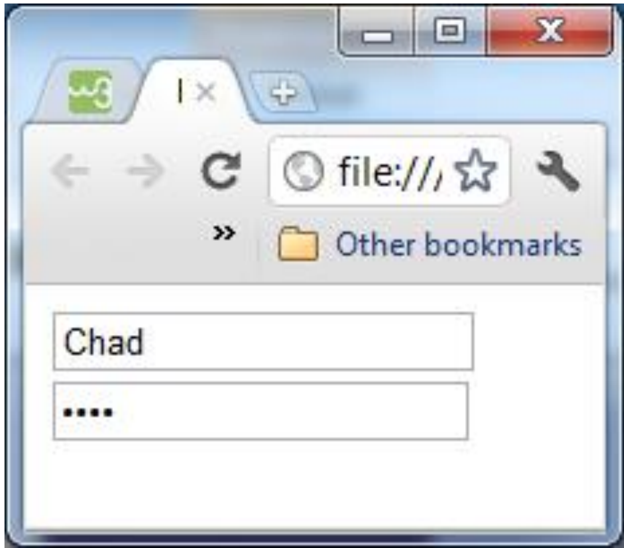
- When viewing a page the form is not visible and it is possible to have multiple forms within the same page
- With form elements you should always give them a name as this will be how you will reference the received data on the server

Password input

- Password input element hides the typed password (Note just because it hides it being typed does **not** mean it is passed across encrypted)

`<input type=password name=“pass” />`

```
<html>
 <head>
  <title>Forms html</title>
 </head>
<body>
 <form>
  <input type="textbox" name="firstName" value="Chad"/><br />
  <input type="password" name="pass" value="Chad" />
 </form>
</body>
</html>
```



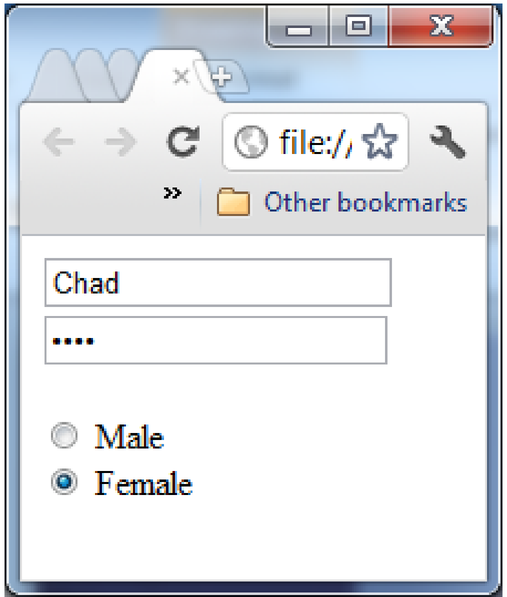
Chad

••••

Radio buttons

- The radio button element allows the user to select **only** one of a given number of choices

```
<form>
<input type="radio" name="sex" value="male" /> Male<br />
<input type="radio" name="sex" value="female" checked/> Female
</form>
```



Chad

•••••

○ Male

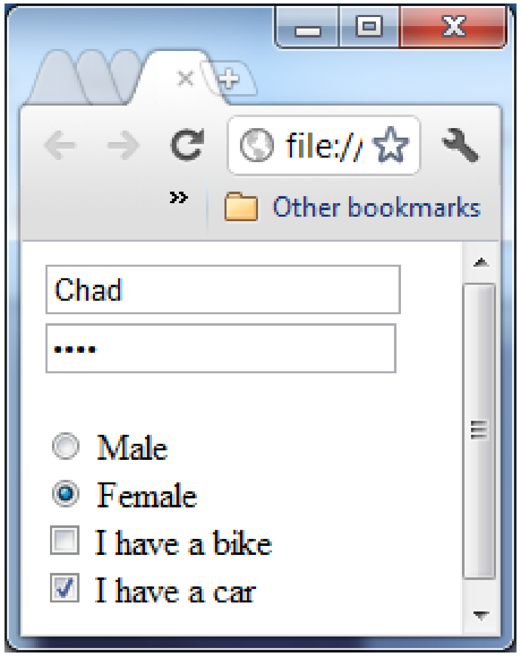
● Female

Sets default value otherwise none are selected

Check boxes

- Allow user to select one or more than one option from a list of choices

```
<form>
<input type="checkbox" name="vehicle" value="Bike" /> I have a bike<br />
<input type="checkbox" name="vehicle" value="Car" checked/> I have a car
</form>
```



Chad

••••

○ Male

● Female

☐ I have a bike

☑ I have a car

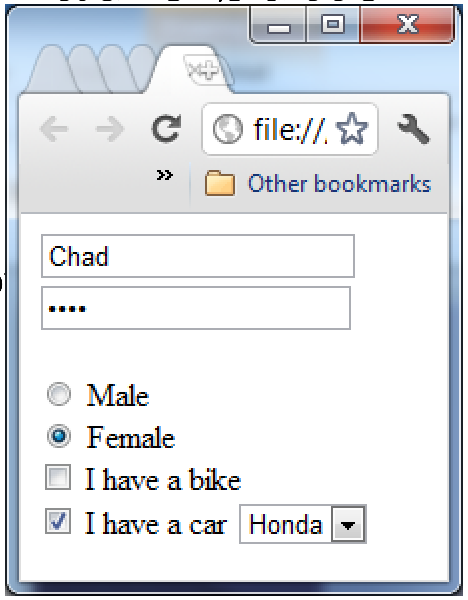
Default some items as checked

Drop down lists

- Allows the user to select only one item from a list of items
- Generally used when there are more choices than could be shown neatly with a radio button

```
<form action="">
<select name="cars">
<option value="volvo">Volvo</option>
<option value="saab">Saab</option>
<option value="honda" selected="selected">Honda</op
<option value="audi">Audi</option
</select>
</form>
```

Sets default value selected



Chad
••••

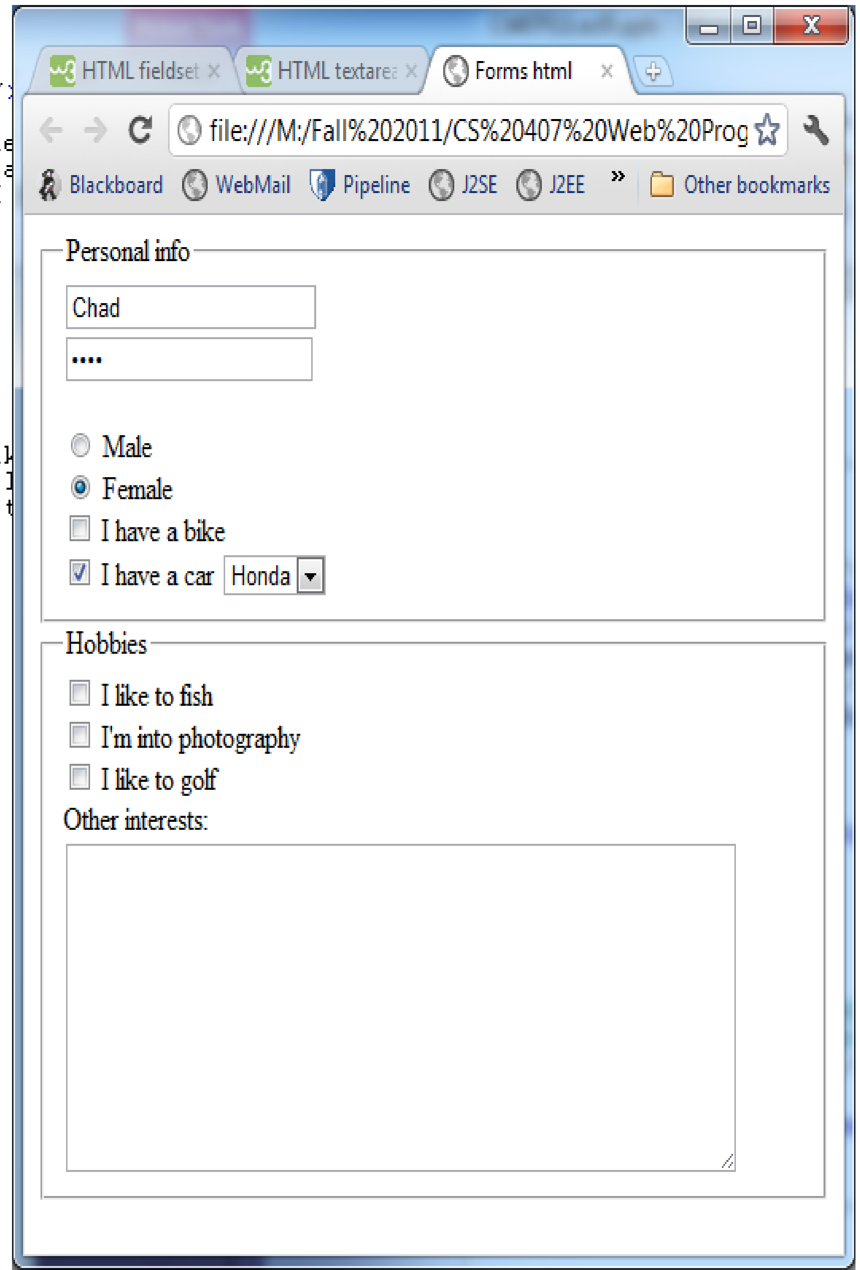
Male
Female
I have a bike
I have a car Honda

Fieldset and legend

- The fieldset element combined with the legend element can be used to group together portions of a form
- Fieldset specifies the grouping
- Legend used within fieldset to label the grouping

Example

```
<form>
  <fieldset>
    <legend>Personal info</legend>
    <input type="textbox" name="firstName" value="Chad"\><br />
    <input type="password" name="pass" value="Chad" \><br /><br /
    <input type="radio" name="sex" value="male" /> Male<br />
    <input type="radio" name="sex" value="female" checked/> Female
    <input type="checkbox" name="vehicle" value="Bike" /> I have a
    <input type="checkbox" name="vehicle" value="Car" checked/> I
    <select name="cars">
      <option value="volvo">Volvo</option>
      <option value="saab">Saab</option>
      <option value="honda" selected="selected">Honda</option>
      <option value="audi">Audi</option>
    </select>
  </fieldset>
  <fieldset>
    <legend>Hobbies</legend>
    <input type="checkbox" name="vehicle" value="Fishing" /> I lik
    <input type="checkbox" name="vehicle" value="Photography" /> I
    <input type="checkbox" name="vehicle" value="Golf" /> I like t
    Other interests:<br/>
    <textarea name="other" cols="50" rows="10"></textarea>
  </fieldset>
</form>
```



Personal info

Chad

••••

○ Male
● Female
☐ I have a bike
☑ I have a car Honda

Hobbies

☐ I like to fish
☐ I'm into photography
☐ I like to golf

Other interests:

Submit button

- A submit button within a form is used to send the form data to the server
- Type=“submit”
- action – specifies the page the form contents should be sent to

```
<form name="input" action="html_form_action.asp" method="get">
Username: <input type="text" name="user" />
<input type="submit" value="Submit" />
</form>
```

Submit method - Get

- Method can be either be either “get” or “post”
- Get
  - Form attributes are sent appended to the URL
  - Pro
    - Allows user to bookmark a page that requires parameters
  - Con
    - there is a limit on the amount of data that can be put in the URL (determined by the browser)
    - data is sent in plain text, easy to manipulate values

Submit method - Post

- “Post” method sends the data as a HTTP Post transaction
- Pro
  - Harder (not impossible) to manipulate posted data
  - No limit on the amount of data that can be put into the post
- Con
  - User cannot bookmark a page where there was posted data, such as a search

Javascript and submit

- Just like other buttons Javascript can act on onClick event to change form elements or even cancel the submit if there is something wrong with the form
- To cancel submit the Javascript executed must “return false;”

Referencing form elements

- In Javascript can simplify accessing elements by using document.myFormName.name rather than document.getElementById(“name”)

- Although both ways are valid, since there may be multiple forms on a page and reason to have elements with the same name it makes it easier to make sure you are getting the element you meant to

Working with arrays (sum.html)

- Create a page with two text boxes (1 for input 1 for output) and a button. In the textbox the user can enter a list of numbers separated by spaces ("1 * 12 + 45 - 3 - 5") and when the button is clicked the sum of the numbers should be output.

Putting it all together (password.html)

- Create a form that contains
  - 2 related radio buttons male/female when you select one the other should deselect
  - 2 related checkboxes have bike and/or car
  - accepts a full name and password
- Check to see if the password contains the person’s initials (ie. If name is Chad A. Williams it can’t contain “CAW”) if it does cancel the submit alerting them of the problem (assume case has to match)
- Otherwise do a GET to the same page (make it easier to see what happened) pay attention to password field and how multiple checked items are passed