First Name Last Name
Student ID: Student ID

CS-90.3 Final Exam

December 15, 2009
Exam ID: Exam ID

Instructions:

- For multiple choice questions, circle the letter of the one best choice unless the question explicitly states that it might have multiple correct answers.
- There is no penalty for guessing, so answer all questions.
- Place drawings where indicated in the question; be sure to put the question number next to your drawing; use pencil rather than ink.
- Unless otherwise indicated, all questions count equally.

1. What does *Core.start()* do?
   A. It receives an object as a parameter, and calls *Core.addEventListener()* to set up the i*nit()* function, defined inside that object, as a listener for the `window.load` event.
   B. It calls the application's *init()* function to initialize the application.
   C. It checks to see if the Core library has been loaded, and displays an error if it has not.
   D. It validates the form data before submitting the form.
   E. It starts the browser's JavaScript processor.

For the next 5 questions use a pencil and draw circles around just the part(s) of each choice that is the kind of item being asked for. For example, in Question 2, if one choice were `var y = f(x);` you would draw a circle around only `f(x)`, not around any of the other parts of the statement.

2. Circle the parts, if any, in each of the following items that are *function calls*.
   - `/@/.test(str);`
   - `Core.addEventListener(email,'change',emailListener);`
   - `var x = { foo:function(){alert('hello');} };`
   - `(function(){alert('hello');})();`
   - `function foo() {alert('hello');}`
3. Now circle the parts, if any, in each of the following items that are *function definitions*:
   - `/@/.test(str);`
   - `Core.addEventListener(email,'change',emailListener);`
   - `var x = { foo:function(){alert('hello');} };`
   - `(function(){alert('hello');})();`
   - `function foo() {alert('hello');}`
4. Now circle the parts, if any, in each of the following items that are *function references*:
   - `/@/.test(str);`
   - `Core.addEventListener(email,'change',emailListener);`
   - `var x = { foo:function(){alert('hello');} };`
   - `(function(){alert('hello');})();`
   - `function foo() {alert('hello');}`

6. Now circle the parts, if any, in each of the following items that are *self-executing functions:*
   - `/@/.test(str);`
   - `Core.addEventListener(email,'change',emailListener);`
   - `var x = { foo:function(){alert('hello');} };`
   - `(function(){alert('hello');})();`
   - `function foo() {alert('hello');}`
7. Now circle the parts, if any, in each of the following items that are *regular expressions*.
   - `/@/.test(str);`
   - `Core.addEventListener(email,'change',emailListener);`
   - `var x = { foo:function(){alert('hello');} };`
   - `(function(){alert('hello');})();`
   - `function foo() {alert('hello');}`
8. What point was made by using both *assignment_05.js* and *assignment_06.js* on the same web page?
   A. Using event *listeners*, you can have two different functions both execute when the `window.load` event occurs.
   B. Using event *handlers*, you can have two different functions both execute when the `window.load` event occurs.
   C. The code in *assignment_06.js* made the code in *assignment_05.js* stop working.
   D. Adding *assignment_06.js* eliminated the need for *core.js*.
   E. Style sheets and JavaScript are the same thing.

Read all of the next three questions before answering any of them.
Given this XHTML:

```
<input id='target' type='text' name='target' />
<span id='target-status' class='initial'>Empty</span>
```

9. `Write three CSS rules that will specify different text and background colors for all elements with classes of 'initial', 'stage-1', and 'stage-2'.` For the initial class, make it red text on a white background; for stage-1 make it black text on a yellow background, and for the stage-2 class make it green text on a white background. Answer here:

10. Complete the definition of the following JavaScript function that initializes the value of a global variable named *targetStatus* with a reference to the DOM object corresponding to the span in the above XHTML code, then initializes a local variable named *target* with a reference to the `input` element in the above XHTML code, and finally sets up a function named *targetListener* as an event listener for `change` events generated from the *target* element. Note: this is only part of the application. Assume the Core library has been loaded and that this function definition is inside the object being passed to *Core.start()* in "the usual way."

```
init: function()
{
  targetStatus =

  var status   =

  Core.
}
```

11. Complete the definition of the event listener for this project. The event listener is to set the class of the *targetStatus* to `initial` if the input field is empty, to `stage-1` if the input field contains between one and three characters (any characters), and to `stage-2` otherwise. The text in the *targetStatus* span is to be 'Empty' if the input field is empty, 'Short' if the input field is 1-3 characters long, and "OK" otherwise. Use a regular expression to test the number of characters.

```
function targetListener(evt)
{
  evt =
  if ( target.value == '')
  {
    targetStatus.firstChild.nodeValue =

    targetStatus.setAttribute(

  }
  else if (
  {

    targetStatus.firstChild.nodeValue =

    targetStatus.setAttribute(

  }
  else
  {




  }
}
```

Assume the XHTML code above is inside a form that begins, `<form method='get' action='scripts/process_form.php'>`. Also assume a user has typed "I am a good student" into the input element. Answer the following questions about what happens when the form is submitted:

12. Will the form data be visible in the URL when the form is submitted?
    A. No, because the POST method was used.
    B. No because the GET method was used.
    C. Yes, because the POST method was used.
    D. Yes because the GET method was used.
    E. No, the form data never appears in the URL regardless of what method is used.
13. Which of the following pieces of code inside *process_form.php* will display the message, "I am a good student" (without needing any additional code)?
    A. `<?php $out = "I am a good student"; ?>`
    B. `<?php echo $_GET['I am a good student']; ?>`
    C. `<?php echo $_GET['target']; ?>`
    D. `<?php echo $_POST['grape-nuts']; ?>`
    E. `<?php echo $_GET[target.value]; ?>`
14. Aside from generating the correct content for web pages, what is the most important consideration in writing PHP scripts for web sites?
    A. Colors
    B. Fonts
    C. Images
    D. Validation
    E. Security
15. Once you have written good JavaScript code to validate form data, do you still need to validate the form data again in PHP? Explain why or why not.