

50 points total

1 (22 pts.). The following is a listing with gaps of the HTML document whose rendering is shown at right. The entire body of the document is a form. It contains a text box with name `name` (for the user's name). It also has a field-set with caption "This is your" encompassing three radio buttons, each with name `nametype`. The value for the first is 'first', the value for the second is 'last', and the value of the third is 'full'. Another field-set, with caption "How much time do you have now?", encompasses a drop-down menu named `time` with three alternatives, named 'lots' (with text `Lots`), 'notMuch' (with text `Not much`), and 'none' (with text `None`). Finally, there is a submit button.

Your name: Fred

How much time do you have now?
Lots

This is your
 First name
 Last name
 Full name

Submit

You must enter your first name and last name, and nothing more.

Done

When this document finishes loading, function `start()` is called. This function assigns a reference to the form to variable `fm`. It then makes the function `verify` the event handler for submit events and makes the function `fastSubmit` the event handler for change events for the drop-down menu. Function `fastSubmit()` checks whether the value for `time` is 'none'; if so, it forces a submission of the form (essentially simulating a click of the submit button).

Function `verify()` copies the value of text box `name` into variable `nm` and searches through the elements of the radio button array `nametype` for the button that is checked; if one is checked, its value is copied to the variable `nmttype`. It then checks whether any of four error conditions hold: (1) no radio button was checked, (2) `nmttype` is 'first' but the value in the text box does not have the pattern of a single name (an initial uppercase letter followed by one or more lowercase letters), (3) `nmtaype` is 'last' but the value in the text box does not fit the pattern of a single name, and (4) `nmttype` is 'full' but the value in the text box does not fit the pattern for a full name (two single names separated by any amount of whitespace). If any of these error conditions holds, `verify()` calls the function `unverified()`, passing it the event that was passed to it and a string containing a message explaining what is wrong.

Function `unverified()` suppresses the submission, makes a new `p` element, assigns the string it was passed as the content of the `p` element, and adds the `p` element to the end of the body of the document. The screenshot above shows the case where the user indicated a full name but entered only a first name. After an attempt to submit, the text at the bottom appeared.

The gaps in the following listing are labeled with Greek letters. After the listing, these letters are repeated along with a description of the code that should file the corresponding gaps. You there supply the missing code.

```
<html>
<head>
  <title>Exam 2, Problem 1</title>

  <script type="text/javascript">
    var fm;

    function start()
    {
      fm = α_____ ;
      β_____ ;
      γ_____ ;
    }

    function fastSubmit( event )
    {
      

δ


    }

    function verify( event )
    {
      var oneName = ε_____ ,
          fullName = ζ_____ ,
          nm = η_____ ,
          nmtypes = θ_____ ,
          ntsize = nmtypes.length,
          nmtype;
```

ι

Continued

Continued from previous page

```
if ( ! nmtype ) {
    unverified( event, "You must check a radio button." );
}
else if ( nmtype == 'full' && ! nm.match( fullName ) ) {
    unverified( event, "You must enter your first " +
        "name and last name, and nothing more." );
}
else if ( nmtype == 'first' && ! nm.match( oneName ) ) {
    unverified( event, "You must enter your first " +
        "name and nothing more." );
}
else if ( nmtype == 'last' && ! nm.match( oneName ) ) {
    unverified( event, "You must enter your last " +
        "name and nothing more." );
}
}

function unverified( event, msg )
{
    K
}

</script>
</head>
<body onload="start()">
<form action="prob2.php" method="post">
    <p>Your name: λ <p>
    μ

```

Continued

Continued from previous page

v

```
<p style="position: absolute; top: 100; left: 185">
  <input id="sbmt" type="submit" value="Submit">
</p>
</form>
</body>
</html>
```

α (1 pt.): A reference to the form element.

Answer: `document.forms[0]`

β (1 pt.): Make function `verify` the listener for submit events.

Answer: `fm.addEventListener("submit", verify, false);`

γ (1 pt.): Make function `fastSubmit` the listener for change events in the drop-down menu, `time`.

Answer: `fm.time.addEventListener("change", fastSubmit, false);`

δ (2.5 pts.): Write the body for `fastSubmit()`. This forces a submission (in effect, clicking the submit button) if the value for the drop-down menu, `time`, is 'none'.

Answer:

```
var tm = fm.time.value;
if ( tm == 'none' )
  document.getElementById( "sbmt" ).click();
```

ϵ (1 pt.): A regular expression that matches a name (an uppercase letter followed by one or more lowercase letters) with nothing before or after it.

Answer: `/^[A-Z][a-z]+$/`

ζ (1.5 pts.): A regular expression that matches a full name: a first followed by a last name, that is, an uppercase letter followed by one or more lowercase letters (i.e., the name pattern), then any amount of whitespace, then the name pattern again. There should be no characters before or after this pattern.

Answer: `/^[A-Z][a-z]+\s+[A-Z][a-z]+$/`

η (1 pt.): The value of the text box named name

Answer: `fm.name.value`

θ (1 pt.): The array of elements corresponding to the radio buttons named nametype

Answer: `fm.nametype`

ι (2 pts.). A loop to find which element of the nmtypes radio buttons was checked. This sets variable nmtype to the value of the checked radio button.

Answer:

```
for ( i=0; i<nsize && ! nmtype; i++ )
    if ( nmtypes[i].checked )
        nmtype = nmtypes[i].value;
```

κ (3 pts.): Write the body of the function `unverified(event, msg)`, where `event` is an `Event` object and `msg` is a string with an error message. This suppresses the submission, makes a new `p` element, assigns `msg` as the content of the `p` element, and adds the `p` element to the end of the body of the document.

Answer:

```
event.preventDefault();
var newP = document.createElement( "p" );
newP.innerHTML = msg;
document.body.appendChild( newP );
```

λ (1 pt.): A text box named name whose width is the width of 18 characters

Answer: `<input type="text" name="name" size="18">`

μ (3 pts.): A field-set with caption “This is your”. It is 5 pixels from the top and 220 pixels from the left margin. It contains the three radio buttons shown in the screenshot, all named ‘nametype’. Their values are, from top to bottom, ‘first’, ‘last’, and ‘full’.

Answer:

```
<fieldset style="position: absolute; top: 5; left: 220">
  <legend>This is your</legend>
  <input type="radio" name="nametype" value="first">
    First name<br />
  <input type="radio" name="nametype" value="last">
    Last name<br />
  <input type="radio" name="nametype" value="full">
    Full name
</fieldset>
```

v (3 pts.): A field-set, with caption “How much time do you have now?”, with a line break between “time” and “do”. It encompasses a drop-down menu named time with three alternatives, named ‘lots’ (with text Lots), ‘notMuch’ (with text Not much), and ‘none’ (with text None). The field-set is 120 pixels wide.

Answer:

```
<fieldset style="width: 120">
  <legend>How much time<br>do you have now?</legend>
  <select name="time" size="1">
    <option value="lots">Lots</option>
    <option value="notMuch">Not much</option>
    <option value="none">None</option>
  </select>
</fieldset>
```

2 (11 pts.). The following listing is the shell for a PHP script that reads one file a line at a time and numbers the lines 1, 2, 3, ... and outputs them to another file. The input file is the file prob2in.txt located in the phpData folder that is a sibling of the document root. The output file is (or will be) the file prob2out.txt in the same folder. If the output file already exists, it should be overwritten. If it does not exist, it will be created. Suppress any errors when you open a file, but, just after opening it, check whether the attempt failed. If so, send a message to the user, output all pending closing tags, and exit. Each line number in the output file should be followed by a space; after the space, the line from the input file occurs. For example, if the contents of the input file are

```
The cat
sat on
the mat.
```

then the contents of the output file should be

```
1 The cat
2 sat on
3 the mat.
```

The boxed part below is where the missing PHP script should be. You can supply it after the listing.

```
<?php
    $DOCUMENT_ROOT = $_SERVER[ 'DOCUMENT_ROOT' ];
?>
<html>
<head>
    <title>Exam 2, Problem 2</title>
</head>
<body>
<?php
    

Missing code (more than will fit in this box)


?>
    <p>The file has been copied</p>
</body>
</html>
```

Answer:

```
@ $fp_in = fopen("$DOCUMENT_ROOT/../../phpData/prob2in.txt", 'rb');
if ( ! $fp_in ) {
    echo "<p>The input file could not be opened.</p>\n".
        "</body>\n</html>";
    exit;
}

@ $fp_out = fopen("$DOCUMENT_ROOT/../../phpData/prob2out.txt", 'wb');
if ( ! $fp_out ) {
    echo "<p>The output file could not be opened.</p>\n".
        "</body>\n<html>";
    exit;
}

while ( ! feof( $fp_in ) ) {
    $line = fgets( $fp_in, 999 );
    fwrite( $fp_out, (++ $line_num)." $line", strlen($line)+2 );
}

fclose( $fp_in );
fclose( $fp_out );
```

3 (7 pts.). The following PHP script with gaps defines three associative arrays, `$master`, `$update1`, and `$update2`, where string keys are associated with integer values. All the keys in `$update1` and all those in `$update2` also occur in `$master`, but there are keys in `$master` that do not occur in `$update1` and others that do not occur in `$update2`. You are to write code that, for each key in `$master`, adds to its associated value the values (if any) associated with the same key in `$update1` and in `$update2`. Do this with a single loop. (The idea is that `$update1` and `$update2` contain amounts to be added to the designated individual's accounts in `$master`.) It should be easy to change the code if several more update files are used.) Write the code that does not depend on the particular elements that happen to be in the two arrays. **Hint:** Attempting to access a value associated with a key that does not exist in the array gives a NULL value, which coerces to 0 as an integer. For the arrays as they are, the output is

```
Ed => 6
Sue => 6
Al => 2
Pam => 6
```

Be sure to clean up after your loop.

```
<?php
    $master = array( 'Ed' => 2, 'Sue' => 3, 'Al' => 1,
                    'Pam' => 4 );
    $update1 = array( 'Pam' => 2, 'Ed' => 1, 'Sue'=> 2 );
    $update2 = array( 'Al' => 1, 'Sue' => 1, 'Ed'=> 3 );
```

Missing code

```
?>
<html>
<head>
    <title>Exam 2, Problem 3</title>
</head>
<body>
<?php
    foreach ( $master as $key => $value )
        echo "$key => $value<br />";
?>
</body>
</html>
```

Write the answer in the next page.

Answer:

```
foreach ( $master as $key => &$value )  
    $value += $update1[$key] + $update2[$key];  
  
unset( $value );  
reset( $master );
```

4 (5 pts.): What factors contribute to the need to scale an e-commerce hardware platform? What are the different ways to scale, and what are their relative advantages? Be brief and to the point, and do not copy verbatim from the text.

5. If you were in charge of e-commerce security for a small to medium size e-commerce operation, what would be the main issues regarding access controls? (This relates both to policy and to implementation.)