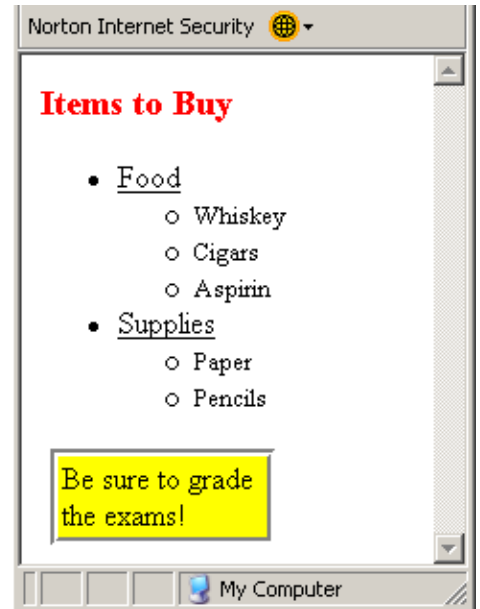
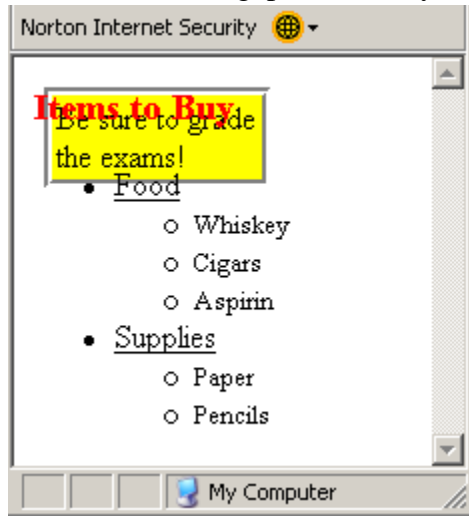


1 (13 pts.) The HTML listed below with gaps is initially rendered as shown on the left.

The box moves down the window and comes to rest below the list, as shown on the right. The box is a paragraph (p element) occupying 60% of the width of the window and floating left. The text in the box is two pixels from the border, which is a groove. A margin of five pixels around the box does not come into play since the box's position is manipulated. The box has a yellow background color and is rendered behind the other elements. The h3 header at the top is red.



In more detail, the box starts ten pixels from the top of the window and moves down five pixels every tenth of a second until it is 200 pixels from the top. Note that outer list items are underlined while the inner list items are not.

This document uses an external stylesheet `prob1.css`, which specifies the that contents of a bulleted list are underlined except for the contents of bulleted lists nested inside other bulleted lists. It also defines the class `box`, which specifies that the element occupies 60% of the width of the window, floats left, has two pixels between its text content and its border, which is a groove, and has a margin of five pixels. And `prob1.css` imports `probla.css`, which is given to you in full, after the listing of the HTML document.

The following is the listing with gaps of the HTML document. The gaps in the listing are labeled with Greek letters. The letters are repeated after the listing with descriptions of the code that should go in the corresponding gaps. You there supply the missing code. You are also asked to write the stylesheet `prob1.css`.

```
<html>
<head>
  <title>Exam 1, Problem 1</title>
  <link rel = "stylesheet" type = "text/css"
        href = "probl.css">
  <script type="text/javascript">
    var offset = 10,
        increment = 5,
        handle;

    function start()
    {
      handle = α_____ ;
    }

    function run()
    {
      offset += increment;
      β_____

      if ( offset >= 200 )
        γ_____
    }
  </script>
</head>
<body onload="start()">
  <p id = "par"
    δ_____>
    Be sure to grade the exams!
  </p>

  <h3 ε_____>Items to Buy</h3>
  <ul>
    <li>Food</li>
    <ul>
      <li>Whiskey</li>
      <li>Cigars</li>
      <li>Aspirin</li>
    </ul>
    <li>Supplies</li>
    <ul>
      <li>Paper</li>
      <li>Pencils</li>
    </ul>
  </ul>
</body>
</html>
```

The following is the listing of stylesheet `probla.css`:

```
p { z-index: -1; background-color: yellow }
```

α (1.5 pts.): Cause the function `run()` to be invoked every tenth of a second.

Answer

```
window.setInterval( "run()", 100 ) // window is optional
```

β (1.5 pts.): Update the location of the paragraph with id `par` so that it is offset pixels from the top of the window.

Answer

```
par.style.top = offset;
```

γ (1.5 pts.): Stop the effects of α , i.e., `run()` will no longer be called.

Answer

```
window.clearInterval( handle ); // window is optional
```

δ (1.5 pts.): Apply class `box` to this element and specify that it is 10 pixels from the top of the window.

Answer

```
class="box" style="position: absolute; top: 10px"
```

ϵ (1 pt.): Specify that the text of this element is red.

Answer

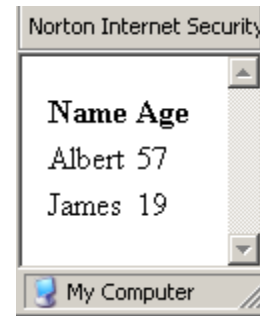
```
style="color: red"
```

Now write the stylesheet `prob1.css`. (6 pts.)

Answer

```
@import url(probla.css); /* 1 pt. */
ul { text-decoration: underline } /* 1.5 pts. */
ul ul { text-decoration: none } /* 1.5 pts. */
.box { width: 60%; float: left; margin: 5px;
padding: 2px; border-style: groove } /* 2 pts. */
```

2 (13 pts.). The JavaScript file listed below with gaps begins execution with a call (when the body of the HTML document is loaded) to function `start()`. This function repeatedly prompts for an input consisting of a name, followed by a comma, then any number of whitespaces, and finally a number consisting of one or two digits. If the input does not match this pattern, an alert box is raised telling the user that the input is in the wrong format. If the input does match, then the string of digits is copied into the associative array `age` at the cell whose key is the name. When the loop is exited (because the user clicked **Cancel**), `age` is passed to the function `outInfo()`, which displays this associative array as a table; an example is shown at right. Note that the pattern does not allow any superfluous characters before the name or after the digits.



The following is the listing of the file with gaps. The gaps in the listing are labeled with Greek letters. The letters are repeated after the listing with descriptions of the code that should go in the corresponding gaps. You there supply the missing code.

```
function start()
{
    var reg = α _____,
        age = β _____,
        info;

    while ( info = prompt( "Name and age in the format\n"
                          + " Name, Age\n"
                          + "Cancel to exit.",
                          "Name, 99" ) )
    {
        if ( γ _____ )
            δ _____
        else
            window.alert( info + ": Wrong format" );

        outInfo( age );
    }
}

function outInfo( age )
{
    document.writeln( "<table>" );
    document.writeln( "<thead>" );
    document.writeln( "<tr><th>Name</th><th>Age</th></tr>" );
    document.writeln( "</thead>" );
    document.writeln( "<tbody>" );

    ε _____
    _____
    _____
    -

    document.writeln( "</tbody>" );
    document.writeln( "</table>" );
}

```

α (4 pts.): The regular expression specifying the pattern consisting of one or more word characters, a comma, zero or more whitespace characters, and one or two digits. Both the string of word characters and the digits are remembered. Also, any string matched by this pattern must be matched completely: the pattern must begin at the beginning of the string and end at the end of the string.

Answer

```
/^(\w+),\s*(\d{1,2})$/
```

β (1 pt.): Create a new associative array.

Answer

```
new Object()
```

γ (1.5 pts.): Check whether the regular expression matches the string just input.

Answer

```
info.match( reg )
```

δ (3 pts.): Copy the string of digits just matched into the associative array `age` at the cell whose key is the string of word characters just matched.

Answer

```
age[ RegExp.$1 ] = RegExp.$2;
```

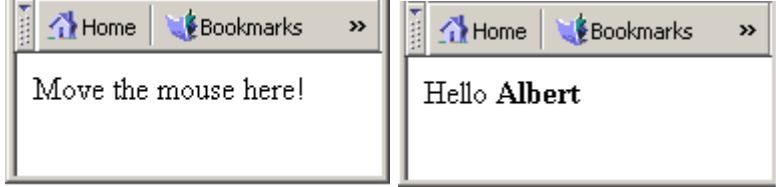
ϵ (4 lines, 3.5 pts.): Output the rows in the body of the table. Use a `for/in` loop.

Answer

```
for ( var name in age ) {  
    document.write( "<tr><td>" + name + "</td>" );  
    document.writeln( "<td>" + age[name] + "</td></tr>" );  
}
```

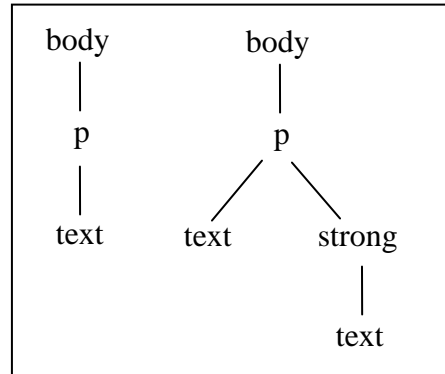
3 (12 points): The HTML document listed below is first rendered as shown below at left. When the user moves the mouse cursor over the text “Move the mouse here!”, (s)he is prompted for his/her name. The original text is then replaced with the text “Hello <name>”, where <name> is the name the user entered and is in bold (cf. the HTML element strong). The screenshot below right is the result when I entered my name.

This is achieved using the DOM (Document Object Model). The initial text is the content of a p element with id par. Immediately below it in the document is a script element



that associates the function greet with it as the listener for mouseover events. Function greet() (defined in the script element in the head) prompts for the user’s name, then creates a p node, a strong node, a text node with the string just input, and a text node with the string “Hello “. It then makes the text node with the name a child of the strong node and the “Hello “ text node then the strong node children of the p node. Finally, it replaces the original p node with the new p node. The part of the original document tree from body on down is shown at left below. This is updated to be as shown at right below.

The following is the listing with gaps of the HTML document. The gaps in the listing are labeled with Greek letters. The letters are repeated after the listing with descriptions of the code that should go in the corresponding gaps. You there supply the missing code.



```

<html>
<head>
  <title>Exam 1, Problem 3</title>
  <script type="text/javascript">
    function greet( event )
    {
      var name = window.prompt( "Your name", "" ),
          newP = α _____,
          strng = β _____,
          strngTxt = γ _____,
          greetTxt = δ _____;

      ε _____
      _____
      _____
      _____
    }
  </script>
</head>
<body>
  <p id="par">Move the mouse here!</p>
  <script type="text/javascript">
    ζ _____
    _____
    _____
  </script>
</body>
</html>

```

α (1.5 pts.): Create a p element node.

Answer

```
document.createElement( "p" )
```

β (1 pt.): Create a strong element node.

Answer

```
document.createElement( "strong" )
```

γ (1 pt.): Create a text node with the string just input.

Answer

```
document.createTextNode( name )
```

δ (1 pt.): Create a text node with “Hello “.

Answer

```
document.createTextNode( "Hello " )
```

ε (5 lines, 5.5 pts.): Now form a tree with the new p node as the root and replace the original p node with this one.

Answer

```
strng.appendChild( strngTxt );
newP.appendChild( greetTxt );
newP.appendChild( strng );
oldP = document.getElementById( "par" );
document.body.replaceChild( newP, oldP );
```

ζ (3 lines, 2 pts.): Make the function greet the listener for mouseover events for the original paragraph. Use the event model of the DOM.

Answer

```
var para =
  window.document.getElementById( "par" );
para.addEventListener( "mouseover", greet, false );
```

4 (6 pts.). Briefly explain what a wireless local area network (WLAN) is. Summarize the standards used WLANs and the situations in which the various standards are typically used. You needn't use exact values; qualitative expressions will do (e.g., "for short distances").

5 (6 pts.). Briefly, how does the Internet benefit the players in an industry value chain?
Give a brief hypothetical example or two.