Project Solutions

Chapter 1

1. Park Bookstore web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="author" content="John Dean">

<title>Park Bookstore</title>

<style>

 h1 {text-align: right;}

 hr {margin-right: 0;}

 div {text-align: right;}

 hr {width: 50%;}

</style>

</head>

<body>

<h1>Park University Bookstore</h1>

<hr>

<div>

 Store Hours: Mon-Fri 9-5<br>

 816-584-6747<br>

 http://www.park.edu/bookstore<br><br>

 Park University<br>

 Bookstore, box 56<br>

 8700 N.W. River Park Drive<br>

 Parkville, MO 64152<br>

</div>

</body>

</html>

1. Galley Menu web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="author" content="John Dean">

<title>Galley Menu</title>

<style>

 hr {margin-left: 0;}

 hr {width: 50%;}

</style>

</head>

<body>

<h1>Galley Menu</h1>

<h4>Today's Special</h4>

<hr>

<h4>Appetizer</h4>

<p>

 Broiled oysters with crumbled bacon and cheddar cheese.

 Topped with hollandaise sauce.

</p>

<h4>Entree</h4>

<p>

 Grilled Alaskan Salmon Fillet. Served with capers, wild rice,

 and fresh vegetables.

</p>

<h4>Dessert</h4>

<p>Poached Pears in Red Wine Sauce.</p>

</body>

</html>

Chapter 2

1. Program Explanation web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="author" content="John Dean">

<title>Program Explanation</title>

</head>

<body>

<h1>Finding the Hypotenuse</h1>

<p>

 The Java code fragment below shows how to calculate the

 hypotenuse of a right triangle.

</p>

<pre><code>public static void main(String[] args)

{

 Scanner stdIn = new Scanner(System.in);

 double base;

 double height;

 double hypotenuse;

 System.out.print("Enter right triangle base: ");

 base = stdIn.nextDouble();

 System.out.print("Enter right triangle height: ");

 height = stdIn.nextDouble();

 hypotenuse = Math.sqrt(base \* base + height \* height);

 System.out.println("Hypotenuse length = " + hypotenuse);

} // end main</code></pre>

<p>

 To see how the program works, try to figure out what the output

 will be if the input values are <kbd>3</kbd> and <kbd>4</kbd>....

</p>

<p>

 The output would be: <samp>Hypotenuse length = 5</samp>.

</p>

<p>

 The hypotenuse is calculated with the help of the Pythagorean

 formula:<br>

 <var>a</var><sup>2</sup> + <var>b</var><sup>2</sup> =

 <var>c</var><sup>2</sup>, where

 <var>c</var> is the hypotenuse, and

 <var>a</var> and <var>b</var> are the triangle's other two sides.

</p>

<p>

 According to the

 <abbr title="Mathematical Association of America">MAA</abbr>,

 the Pythagorean formula is one of the <strong>top 10</strong>

 all-time great math formulas. In his 520 BC bestseller

 <cite>Fifty Shades of Gray Matter</cite>, Pythagoras

 himself says it best - <q>Yeah, it's a pretty cool formula.</q>

</p>

</body>

</html>

Chapter 3

1. Societal Quotes web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="author" content="John Dean">

<title>CSS Practice</title>

<style>

 .bold {font-weight: bold;}

 .strike-through {text-decoration: line-through;}

 p.gray-background {background-color: #cccccc;}

 h1, h2, h3, h4, h5, h6 {

 text-align: center;

 text-transform: uppercase;

 }

 p {

 padding: 5px;

 margin-left: 50px;

 margin-right: 50px;

 border-style: dashed;

 border-color: red;

 font-family: "Times New Roman", serif;

 text-indent: 10px;

 text-align: justify;

 }

</style>

</head>

<body>

...

</body>

</html>

1. Newspaper web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="author" content="John Dean">

<title>The Parkville Times</title>

<style>

 body {background-color: cyan;}

 h1 {

 text-align: center;

 font-style: italic;

 color: blue;

 }

 h3 {text-align: center;}

 .bold {font-weight: bold;}

 .silver {color: hsl(0,0%,75%);} /\* 1st argument is irrelevant \*/

 .fuchsia {color: #ff00ff;}

 .bright-orange {color: rgb(255,131,0);}

 .important {

 font-weight: bold;

 text-decoration: underline;

 }

 .rates {font-family: "Courer New", monospace;}

</style>

</head>

<body>

<h1>The Parkville Times<sup>&reg;</sup></h1>

<hr>

<h3>Late-Breaking, Investigative, Sensationalist</h3>

<div>

 <span class="bold">MAIN OFFICE:</span><br>

 111 Main<br>

 Parkville, MO 64152

</div>

<p>

 Enjoy a wide variety of news stories, comic strips, and advertisements, in these

 vibrant colors:<br>

 <span class="silver">Silver for news stories.</span>

 <span class="fuchsia">Fuchsia for comic strips.</span>

 <span class="bright-orange">Bright orange for advertisements.</span>

</p>

<div>

 <span class="important">Subscription Information:</span><br>

 Call: (816) 741-2000<br>

 Hours: Mon-Sat 8-5

</div>

<br>

<div>

 <span class="important">Rates</span>:<br>

 <span class="rates">

 Every Day..............$2.00<br>

 Sunday Only............$3.00<br>

 Web Only...............$2.50

 </span>

</div>

</body>

</html>

Chapter 4

1. Wealth web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name= "author" content="John Dean">

<title>Retirement Wealth</title>

<style>

 body {background-color: blanchedalmond;}

 header {

 color: white;

 background-color: darkblue;

 }

 section > h2 {

 color: white;

 background-color: blueviolet;

 }

 aside > h3 {margin: 0;}

 aside {

 background-color: white;

 float: right;

 margin: 5px;

 padding: 10px;

 }

 a {color: white; text-decoration: none;}

 a:hover {text-decoration: underline;}

</style>

</head>

<body>

<aside>

 <h3>TNN News, Tuesday March 6, 8pm EST</h3>

 Author Robert Strausser presents "Survival of the Wealthiest"

</aside>

<header>

 <h1>The American Dream</h1>

 <h2>Work hard and become wealthy? Maybe so, but work alone might not

 cut it. The following graphs<sup><a href="#disclaimer">[1]</a></sup>

 show various paths to prosperity.</h2>

</header>

<article>

 <section>

 <h2 id="education">Education</h2>

 <p>

 More advanced schooling helps. If you can avoid the crushing

 debt of student loans, all the better. Try to find a major that

 you're good at and that is good to you.

 </p>

 <figure>

 <img src="educationWealth.png"

 alt="education's affect on retirement wealth">

 </figure>

 </section>

 <section>

 <h2 id="birthright">Birthright</h2>

 <p>

 For accruing wealth, what's even better than education?

 Being born wealthy. In the graph below, note how the birthright

 high enders are able to become more wealthy than the education

 high enders. So before being born, shop around.

 </p>

 <figure>

 <img src="birthWealth.png"

 alt="birth wealth's affect on retirement wealth">

 </figure>

 </section>

 <p id="disclaimer">

 1. Although the graphs' data is fabricated, the trends are real.

 See Lisa Keister's "Getting Rich: America’s New Rich and How They

 Got That Way" and Pew Charitable Trusts' "Economic Mobility Project."

 </p>

</article>

</body>

</html>

Chapter 5

1. Food Pyramid web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="author" content="John Dean">

<title>Food Pyramid</title>

<style>

 body {display: flex; justify-content: center;}

 table, tbody > tr > \* {border: thin solid;}

 thead {

 background-color: red;

 color: white;

 border: none;

 }

 caption {

 font-size: xx-large;

 padding: 10px;

 }

 th, td {padding: 10px;}

 tbody th {text-align: left;}

 .italic {

 font-weight: normal;

 font-style: italic;

 }

 td {text-align: center;}

</style>

</head>

<body>

<table>

 <caption>The Food Pyramid</caption>

 <thead>

 <tr>

 <th></th><th colspan="2">Serving Guidelines</th>

 </tr>

 <tr>

 <th>Food Group</th>

 <th>2005-2010</th>

 <th>2011-present</th>

 </tr>

 </thead>

 <tbody>

 <tr>

 <th>Milk Group<br>

 <span class="italic">(Milk, Yogurt, Cheese)</span></th>

 <td>2-3</td>

 <td>3-4</td>

 </tr>

 <tr>

 <th>Meat Group<br>

 <span class="italic">(Beef, Poultry, Fish, Beans)</span></th>

 <td>2-3</td>

 <td>5-6</td>

 </tr>

 <tr>

 <th>Fruits</th>

 <td>2-4</td>

 <td>2-4</td>

 </tr>

 <tr>

 <th>Vegetables</th>

 <td>3-5</td>

 <td>1-2</td>

 </tr>

 <tr>

 <th>Heavily Salted Snacks</th>

 <td>1-2</td>

 <td>3-5</td>

 </tr>

 </tbody>

</table>

</body>

</html>

1. Gymnastics web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="author" content="John Dean">

<title>Gymnastics</title>

<style>

 body {background-color: lightcyan;}

 /\* EXTRA CREDIT \*/

 body {

 position: relative;

 left: 20px; top: 20px;

 /\* alternative to using left and right:

 margin: 20px; \*/

 }

 \* {margin: 0;}

 header {

 position: absolute;

 top: 0; left: 0;

 width: 500px; height: 85px;

 background-color: darkblue;

 color: gold;

 font-style: italic;

 text-align: center;

 }

 section {

 position: absolute;

 top: 100px;

 /\* the following is unnecessary, but helps with maintenance \*/

 width: 240px; height: 250px;

 }

 section.left {left: 10px;}

 section.right {left: 260px;}

 ul {list-style: circle;}

 figure {

 position: absolute;

 top: 290px; left: 0;

 }

 figcaption {text-align: center;}

</style>

</head>

<body>

<header>

 <h1>Gymnastics</h1>

 <h1>Go for the gold!</h1>

</header>

<section class="left">

 <h3>Greatest Female Gymnasts</h3>

 <ul>

 <li>

 Rumania:

 <ol>

 <li>Nadia Comaneci</li>

 </ol>

 </li>

 <li>

 Russia:

 <ol start="2"> <!-- EXTRA CREDIT -->

 <li>Olga Korbut</li>

 <li>Larisa Latynina</li>

 </ol>

 </li>

 <li>

 United States:

 <ol start="4"> <!-- EXTRA CREDIT -->

 <li>Gabby Douglas</li>

 <li>Mary Lou Retton</li>

 </ol>

 </li>

 </ul>

</section>

<section class="right">

 <h3>Greatest Male Gymnasts</h3>

 <ul>

 <li>

 Russia:

 <ol>

 <li>Vitaly Scherbo</li>

 <li>Alexei Nemov</li>

 <li>Nikolai Andrianov</li>

 </ol>

 </li>

 <li>

 Japan:

 <ol start="4"> <!-- EXTRA CREDIT -->

 <li>Sawao Kato</li>

 </ol>

 </li>

 </ul>

</section>

<figure>

 <img src="magnificent7.jpg"

 alt="Uneven Bars" width="500" height="327">

 <figcaption>Flying Through the Air with the Greatest of Ease

 </figcaption>

</figure>

</body>

</html>

Chapter 6

1. World Weather web page.

<!DOCTYPE html>

<html lang="en">

<head>

<title>World Weather</title>

<meta name="author" content="John Dean">

<link rel="icon" href="../images/jd.gif">

<style>

 h3 > a:link {text-decoration: none;}

 h3 > a:hover {background-color: silver;}

 .table {display: table;}

 .row {display: table-row;}

 .row > :nth-child(1) {

 display: table-cell;

 font-size: x-large;

 text-align: right;

 vertical-align: middle;

 }

 .row > :nth-child(2) {

 display: table-cell;

 padding: 20px;

 }

</style>

</head>

<body>

<h1 id="top">World Weather</h1>

<h3>

 <a href="#africa">Africa</a> |

 <a href="#australia">Australia</a> |

 <a href="#south-america">South America</a>

</h3>

<div>

 <img src="../images/lightning.jpg" width="640" height="350" alt="lightning">

</div>

<div class="table">

 <div id="africa" class="row">

 <span>Africa:</span>

 <span>

 <a href="http://www.holiday-weather.com/dakar/">

 Dakar, Senegal

 </a>

 <br>

 <a href="http://www.holiday-weather.com/johannesburg/">

 Johannesburg, South Africa

 </a>

 <br>

 <a href="http://www.holiday-weather.com/nairobi/">

 Nairobi, Kenya

 </a>

 </span>

 </div>

 <div id="australia" class="row">

 <span>Australia:</span>

 <span>

 <a href="http://www.holiday-weather.com/brisbane/">

 Brisbane, Australia

 </a>

 <br>

 <a href="http://www.holiday-weather.com/great\_barrier\_reef/">

 Great Barrier Reef, Australia

 </a>

 </span>

 </div>

 <div id="south-america" class="row">

 <span>South America:</span>

 <span>

 <a href="http://www.holiday-weather.com/santiago/">

 Santiago, Chile

 </a>

 <br>

 <a href="http://www.holiday-weather.com/rio\_de\_janeiro/">

 Rio de Janeiro, Brazil

 </a>

 </span>

 </div>

</div>

</body>

</html>

Chapter 7

1. Rowing web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="author" content="John Dean">

<title>Rowing</title>

<link rel="icon" href="../images/jd.gif">

<style>

 body {background-image: url(../images/ocean.jpg);}

</style>

</head>

<body>

<h1>World Rowing Federation</h1>

<p>

 This page tells you everything you need to know about rowing.

 Find a boat. Find a body of water. Put the boat in the water. Row.

</p>

<h3>

 This first "Row, Row, Row Your Boat" sound clip starts

 automatically and plays continuously (unless you pause it):

</h3>

<audio autoplay controls loop>

 <source src="../audio/rowYourBoat.mp3" type="audio/mpeg">

 <source src="../audio/rowYourBoat.ogg" type="audio/ogg">

</audio>

<h3>

 Try to start this second sound clip between the first clip's

 "boat" and "gently," so the two clips form a mellifluous harmony

 that plays continuously:

</h3>

<!-- The following preload is unnecessary because the first audio

 element loads the audio file as soon as the web page loads. -->

<audio preload controls loop>

 <source src="../audio/rowYourBoat.mp3" type="audio/mpeg">

 <source src="../audio/rowYourBoat.ogg" type="audio/ogg">

</audio>

<!-- EXTRA CREDIT -->

<h3>Rowing Video:</h3>

<iframe width="420" height="315"

 src="http://www.youtube.com/embed/A8GTgbHYMyI?rel=0"></iframe>

</body>

</html>

Chapter 8

1. Compass Game web page.

<!DOCTYPE html>

<html lang="en">

<head>

<title>Compass Game</title>

<meta name="author" content="John Dean">

<link rel="icon" href="../images/jd.gif">

<style>

 .table {display: table;}

 .table > \* {display: table-row;}

 .table > \* > \* {

 display: table-cell;

 text-align: center;

 font-size: 2em;

 font-weight: 800;

 margin: 15px;

 color: red;

 }

 .arrow {color: blue;}

 .italics {font-style: italic;}

 .diagonal {transform: rotate(-45deg);}

 /\* for older browsers: \*/

 /\*

 .diagonal {

 transform: rotate(-45deg);

 -webkit-transform: rotate(-45deg);

 -moz-transform: rotate(-45deg);

 }

 \*/

</style>

<script>

 // This function erases the compass's directions and

 // activates the text boxes and update button.

 function restartCompass(form) {

 form.reset();

 document.getElementById("top").innerHTML = "";

 document.getElementById("left").innerHTML = "";

 document.getElementById("right").innerHTML = "";

 document.getElementById("bottom").innerHTML = "";

 form.elements["topBox"].disabled = false;

 form.elements["leftBox"].disabled = false;

 form.elements["rightBox"].disabled = false;

 form.elements["bottomBox"].disabled = false;

 form.elements["update"].disabled = false;

 } // end restartCompass

 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 // This function copies the text box values to the compass's

 // directions and disables the text boxes and update button.

 function updateCompass(form) {

 document.getElementById("top").innerHTML =

 form.elements["topBox"].value;

 document.getElementById("left").innerHTML =

 form.elements["leftBox"].value;

 document.getElementById("right").innerHTML =

 form.elements["rightBox"].value;

 document.getElementById("bottom").innerHTML =

 form.elements["bottomBox"].value;

 form.elements["topBox"].disabled = true;

 form.elements["leftBox"].disabled = true;

 form.elements["rightBox"].disabled = true;

 form.elements["bottomBox"].disabled = true;

 form.elements["update"].disabled = true;

 } // end updateCompass

 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 // This function swaps the direction letters for the

 // 2 passed-in positions.

 function swap(direction1, direction2) {

 var temp; // holds direction during swap process

 temp = document.getElementById(direction1).innerHTML;

 document.getElementById(direction1).innerHTML =

 document.getElementById(direction2).innerHTML;

 document.getElementById(direction2).innerHTML = temp;

 } // end swap

 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 // EXTRA CREDIT:

 // This function randomly repositions the direction letters.

 function shuffle(form) {

 var directions = // compass directions

 ["N", "S", "E", "W"];

 var j; // index of swapped element

 var temp; // helps with swap

 for (var i=0; i<directions.length; i++) {

 j = Math.floor(Math.random() \* directions.length);

 temp = directions[i];

 directions[i] = directions[j];

 directions[j] = temp;

 } // end for

 document.getElementById("top").innerHTML = directions[0];

 document.getElementById("left").innerHTML = directions[1];

 document.getElementById("right").innerHTML = directions[2];

 document.getElementById("bottom").innerHTML = directions[3];

 } // end shuffle

</script>

</head>

<body>

<noscript>

 <p>

 This web page uses JavaScript. For proper results,

 you must use a web browser with JavaScript enabled.

 </p>

</noscript>

<h1>Compass Game</h1>

<p>

 The direction letters on the "compass" are scrambled.

 Press the arrows to properly position the direction letters.

</p>

<form>

 <div class="table">

 <div>

 <input type="button" class="arrow diagonal" value="&harr;"

 onclick = "swap('top', 'left');">

 <span id="top">E</span>

 <span></span>

 </div>

 <div>

 <span id="left">S</span>

 <input type="button" class="arrow" value="&harr;"

 onclick="swap('left', 'right');">

 <span id="right">N</span>

 </div>

 <div>

 <span></span>

 <span id="bottom">W</span>

 <input type="button" class="arrow diagonal" value="&harr;"

 onclick="swap('bottom', 'right');">

 </div>

 </div> <!-- end table -->

 <br><br>

 <p>

 To restart the game, press <span class="italics">Restart</span>,

 enter N, S, E, and W in any order in the boxes below, and then

 press <span class="italics">Update</span>.

 </p>

 <input type="button" value="Restart"

 onclick="restartCompass(this.form);">

 <br><br>

 Top:

 <input type="text" id="topBox" size="1" maxlength="1" disabled>

 Left:

 <input type="text" id="leftBox" size="1" maxlength="1" disabled>

 Right:

 <input type="text" id="rightBox" size="1" maxlength="1" disabled>

 Bottom:

 <input type="text" id="bottomBox" size="1" maxlength="1" disabled>

 <br><br>

 <input type="button" id="update" value="Update" disabled

 onclick="updateCompass(this.form);">

 <!-- EXTRA CREDIT:

 <input type="button" value="Shuffle"

 onclick="shuffle(this.form);">

 -->

</form>

</body>

</html>

Chapter 9

1. Roots Calculator web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="author" content="John Dean">

<title>Quadratic Equation Roots Calculator</title>

<style>

 input:focus:invalid {border-color: red;}

</style>

<script>

 // This function calculates and displays the quadratic

 // roots of a polynomial equation.

 function calculateRoots(form) {

 var aBox, bBox, cBox; // text box controls

 var aStr, bStr, cStr; // string values from the text boxes

 var a, b, c; // numeric values from the text boxes

 var error = ""; // error message for alert

 var x1, x2; // the polynomial's roots

 var soln; // solution message at the bottom

 aBox = form.elements["a"];

 bBox = form.elements["b"];

 cBox = form.elements["c"];

 aStr = aBox.value;

 bStr = bBox.value;

 cStr = cBox.value;

 if (!aBox.checkValidity() || aStr === "0") {

 error += "Invalid entry for a." +

 " Must be a non-zero integer between -99 and +99.\n";

 /\* EXTRA CREDIT: \*/

 aBox.style.borderColor = "red";

 }

 if (!bBox.checkValidity()) {

 error += "Invalid entry for b." +

 " Must be an integer between -99 and +99.\n";

 /\* EXTRA CREDIT: \*/

 bBox.style.borderColor = "red";

 }

 if (!cBox.checkValidity()) {

 error += "Invalid entry for c." +

 " Must be an integer between -99 and +99.\n";

 /\* EXTRA CREDIT: \*/

 cBox.style.borderColor = "red";

 }

 if (error !== "") {

 document.getElementById("soln").innerHTML = "";

 // Nice, but not required: Omit trailing \n.

 alert(error.substring(0, error.length - 1));

 }

 else {

 /\* EXTRA CREDIT: \*/

 aBox.style.borderColor = "";

 bBox.style.borderColor = "";

 cBox.style.borderColor = "";

 a = parseInt(aStr);

 b = parseInt(bStr);

 c = parseInt(cStr);

 soln = "Solution: ";

 if (b \* b < 4 \* a \* c) {

 soln += "<var>x</var>'s roots are imaginary";

 }

 else if (b \* b == 4 \* a \* c) {

 x1 = -b / (2 \* a);

 soln += "<var>x</var> = " + x1.toFixed(3);

 }

 else {

 x1 = (-b + Math.sqrt(b \* b - 4 \* a \* c)) / (2 \* a);

 x2 = (-b - Math.sqrt(b \* b - 4 \* a \* c)) / (2 \* a);

 soln += "<var>x</var> = " + x1.toFixed(3) +

 ", <var>x</var> = " + x2.toFixed(3);

 }

 document.getElementById("soln").innerHTML = soln;

 } // end else

 } // end calculateRoots

</script>

</head>

<body>

<h2>Quadratic Equation Roots Calculator</h2>

<p>

 This web page finds the roots of a polynomial equation of the

 following form:

 <br>

 <var>a</var><var>x</var><sup>2</sup>

 + <var>b</var><var>x</var> + <var>c</var> = 0

</p>

<p>

 <var>a</var>, <var>b</var>, and <var>c</var> must be integers

 between -99 and +99, and <var>a</var> must be non-zero.

</p>

<form>

 <label for="a">a:</label>

 <input type="number" id="a"

 min="-99" max="99" step="1" required>

 &nbsp;&nbsp;

 <label for="b">b:</label>

 <input type="number" id="b"

 min="-99" max="99" step="1" required>

 &nbsp;&nbsp;

 <label for="c">c:</label>

 <input type="number" id="c"

 min="-99" max="99" step="1" required>

 <br><br>

 <input type="button" value="Calculate roots"

 onclick="calculateRoots(this.form);">

</form>

<p id="soln"></p>

</body>

</html>

Chapter 10

1. Story Generator web page.

storyGenerator.html:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Story Generator</title>

<meta name="author" content="John Dean">

<script src="storyGenerator.js"></script>

<style>

 .buttonLabel {font-family: Tahoma, sans-serif;}

 .table {display: table;}

 .row {display: table-row;}

 .row > :nth-child(1) {

 display: table-cell;

 text-align: right;

 padding-right: 10px;

 }

 .row > :nth-child(2) {

 display: table-cell;

 }

 .hidden {display: none;}

 #page2 {

 font-family: "Comic Sans MS", cursive;

 color: blue;

 }

 .red {color: red;}

 .blue {color: blue;}

</style>

</head>

<body>

<section id="page1" class="">

 <h1>Story Generator</h1>

 <p>

 Enter appropriate values in all of the text boxes.

 Then click the <span class="buttonLabel">Done</span> button to

 generate a story with your entered values plugged into it.

 </p>

 <form id="form1" class="table">

 <div class="row">

 <label for="word1">noun: </label>

 <input id="word1" type="text" size="25">

 </div>

 <div class="row">

 <label for="word2">past tense verb: </label>

 <input id="word2" type="text" size="25">

 </div>

 <div class="row">

 <label for="word3">adjective: </label>

 <input id="word3" type="text" size="25">

 </div>

 <div class="row">

 <label for="word4">past tense verb: </label>

 <input id="word4" type="text" size="25">

 </div>

 <div class="row">

 <label for="word5">adjective: </label>

 <input id="word5" type="text" size="25">

 </div>

 </form>

 <br>

 <input type="button" form="form1" value="Done"

 onclick="process(this.form)">

 <input type="reset" form="form1" value="Reset">

</section>

<section id="page2" class="hidden">

 <p id="story">

 It was a dark and stormy night. A shot rang out! A

 <span class=""></span> slammed. The maid <span class=""></span>.

 Suddenly, a <span class=""></span> ship appeared on the horizon!

 While millions of people <span class=""></span>, the king lived in

 luxury. Meanwhile, on a <span class=""></span> farm in Kansas,

 a boy was growing up.

 </p>

 <p>The End</p>

 <br>

 <input type="button" value="Return" onclick="returnToPage1();">

</section>

</body>

</html>

storyGenerator.js:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* storyGenerator.js

\* John Dean

\*

\* This file contains functions that support the

\* story generator web page.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// This function processes the story generator's Done button

// being clicked.

function process(form) {

 var userEntry; // one of the text box entries

 var storyHole; // one of the holes in the story

 // Loop through all the form controls except for the

 // two buttons at the bottom.

 for (let i=0; i<form.elements.length-2; i++) {

 userEntry = form.elements[i];

 storyHole = document.getElementById("story").children[i];

 fillStoryHole(userEntry, storyHole);

 } // end for

 document.getElementById("page1").className = "hidden";

 document.getElementById("page2").className = "";

} // end process

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// If the user entry is empty, fill in the story hole with a red

// default message. If non-empty, fill it in with the user entry.

function fillStoryHole(userEntry, storyHole) {

 if (userEntry.value === "") {

 storyHole.innerHTML = "&lt;missing text&gt;"

 storyHole.className = "red";

 }

 else {

 storyHole.innerHTML = userEntry.value;

 storyHole.className = "blue";

 }

} // end fillStoryHole

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// This function hides the story section and restores

// the entry form.

function returnToPage1() {

 document.getElementById("page2").className = "hidden";

 document.getElementById("page1").className = "";

} // end returnToPage1

Chapter 11

1. Statistics Calculator web page.

statistics.html:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Statistics Calculator</title>

<meta name="author" content="John Dean">

<script src="statistics.js"></script>

</head>

<body onload = "numberList = new NumberList();">

<h1>Statistics Calculator</h1>

<h3>Enter a list of numbers separated by spaces.</h3>

<form id="form1" class="table">

 <label for="numList">Number list:</label>

 <input id="numList" type="text" size="20"

 onchange="numberList.readNumbers(this.form);">

 <br><br>

 <input type="button" value="Mean"

 onclick="numberList.calculate(this);">

 &nbsp;&nbsp;&nbsp;

 <input type="button" value="Variance"

 onclick="numberList.calculate(this);">

 &nbsp;&nbsp;&nbsp;

 <input type="button" value="Standard Deviation"

 onclick="numberList.calculate(this);">

</form>

<br>

<output id="result" form="form1"></output>

</body>

</html>

statistics.js:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* statistics.js

\* John Dean

\*

\* The user enters a list of numbers in a text box

\* and then clicks a button to calculate either the mean,

\* variance, or standard deviation.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

class NumberList {

 constructor() {

 this.numbers = null; // array that stores user-entered numbers

 this.validNumbers = false;

 }

 // After user changes the text box content, read the numbers

 // and store in an array.

 readNumbers(form) {

 var number;

 this.validNumbers = true;

 this.numbers = form.elements["numList"].value.split(" ");

 for (let i=0; i<this.numbers.length && this.validNumbers; i++) {

 number = parseFloat(this.numbers[i]);

 if (isNaN(number)) {

 form.elements["result"].value =

 "The input is invalid. You must enter valid numbers" +

 " in the Number List box.";

 this.validNumbers = false;

 }

 else {

 this.numbers[i] = number;

 }

 }

 } // end readNumbers

 // Handle the user clicking one of the buttons.

 calculate(button) {

 var form;

 var numbersStr;

 var result;

 form = button.form;

 if (this.numbers == null || this.numbers.length == 0) {

 form.elements["output"].value =

 "The Number list box is empty. You must enter numbers in it.";

 }

 else if (this.validNumbers) {

 numbersStr = this.numbers[0];

 for (let i=1; i<this.numbers.length; i++) {

 numbersStr += ", " + this.numbers[i];

 }

 if (button.value == "Mean") {

 form.elements["result"].value =

 "mean[" + numbersStr + "] = " +

 this.getMean().toFixed(2);

 }

 if (button.value == "Variance") {

 form.elements["result"].value =

 "variance[" + numbersStr + "] = " +

 this.getVariance().toFixed(2);

 }

 if (button.value == "Standard Deviation") {

 form.elements["result"].value =

 "stdDev[" + numbersStr + "] = " +

 this.getStdDeviation().toFixed(2);

 }

 } // ene else

 } // end calculate

 // Calculate the mean and return it.

 getMean() {

 var sum = 0;

 for (let i=0; i<this.numbers.length; i++) {

 sum += this.numbers[i];

 }

 return sum / this.numbers.length;

 } // end getMean

 // Calculate the variance and return it.

 getVariance() {

 var sum = 0; // sum of (numbers[i] - mean)^2

 var mean;

 mean = this.getMean();

 for (let i=0; i<this.numbers.length; i++) {

 sum += Math.pow(this.numbers[i] - mean, 2);

 }

 return sum / this.numbers.length;

 } // end getVariance

 // Calculate the standard deviation and return it.

 getStdDeviation() {

 return Math.sqrt(this.getVariance());

 } // end getStdDeviation

} // end class NumberList

Chapter 12

1. Combination Lock web page.

combinationLock.html:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name= "author" content="John Dean">

<title>Combination Lock</title>

<style>

 .centered {

 width: 500px;

 text-align: center;

 }

 #target {width: 3em;} /\* EXTRA CREDIT \*/

</style>

<script src="combinationLock.js"></script>

</head>

<body onload="initializeCanvas();">

<canvas id="canvas" width="500" height="370">

</canvas>

<br><br>

<form class="centered">

 <input type="button" value="Nudge left"

 onclick="nudgeLeft();">

 <input type="button" value="Nudge right"

 onclick="nudgeRight();">

 <!-- EXTRA CREDIT: -->

 <br><br>

 Target:

 <input type="number" id="target" value="0"

 min="0" max="39" step="1" required>

 &nbsp;&nbsp;

 <input type="button" value="Left to target"

 onclick="leftToTarget(this.form);">

 <input type="button" value="Right to target"

 onclick="rightToTarget(this.form);">

</form>

</body>

</html>

combinationLock.js:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* combinationLock.js

\* John Dean

\*

\* This file handles drawing a combination lock and

\* having it move.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// x and y coordinates for the face's center point.

const CENTER\_X = 250; // center of dial x position

const CENTER\_Y = 200; // center of dial y position

const CASING\_RADIUS = 160; // casing for the dial

const DIAL\_RADIUS = 120; // dial shows hash marks & numbers

const KNOB\_RADIUS = 45; // the turning mechanism (if real)

const POINTER\_HEIGHT = 16; // height of triangle at top

const NUM\_OF\_TICK\_MARKS = 40; // tick indicate dial positions

const TICK\_WIDTH = 4; // width of each tick mark

var ctx; // the canvas object's context

// EXTRA CREDIT:

var currentTick = 0; // number at top of dial

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// This function draws the combination lock.

function initializeCanvas() {

 var canvas; // the canvas element

 canvas = document.getElementById("canvas");

 ctx = canvas.getContext("2d");

 drawCasing();

 drawDial(0);

} // end initializeCanvas

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// This function draws the lock's casing.

function drawCasing() {

 ctx.fillStyle = "silver";

 ctx.strokeStyle = "black";

 ctx.lineWidth = 1;

 ctx.beginPath();

 ctx.arc(CENTER\_X, CENTER\_Y, CASING\_RADIUS, 0, 2 \* Math.PI);

 ctx.fill();

 ctx.stroke();

 // Draw line that surrounds the dial

 ctx.beginPath();

 ctx.arc(CENTER\_X, CENTER\_Y, DIAL\_RADIUS + 4,

 -.48 \* Math.PI, 1.48 \* Math.PI);

 ctx.stroke();

 // Draw dial's pointer

 ctx.fillStyle = "red";

 ctx.beginPath();

 ctx.moveTo(CENTER\_X, CENTER\_Y - DIAL\_RADIUS);

 ctx.lineTo(CENTER\_X + POINTER\_HEIGHT,

 CENTER\_Y - DIAL\_RADIUS - POINTER\_HEIGHT);

 ctx.lineTo(CENTER\_X - POINTER\_HEIGHT,

 CENTER\_Y - DIAL\_RADIUS - POINTER\_HEIGHT);

 ctx.fill();

} // end drawCasing

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// This function draws the lock's dial.

function drawDial(adjustedTicks) {

 ctx.translate(CENTER\_X, CENTER\_Y);

 // Use passed-in ticks to adjust current rotatation.

 ctx.rotate(adjustedTicks \* 2 \* Math.PI / NUM\_OF\_TICK\_MARKS);

 // draw dial background

 ctx.beginPath();

 ctx.fillStyle = "black";

 ctx.arc(0, 0, DIAL\_RADIUS, 0, 2 \* Math.PI);

 ctx.fill();

 // draw knob

 ctx.beginPath();

 ctx.fillStyle = "darkslategray";

 ctx.arc(0, 0, KNOB\_RADIUS, 0, 2 \* Math.PI);

 ctx.fill();

 // Draw center label

 ctx.fillStyle = "white";

 ctx.textAlign = "center";

 ctx.textBaseline = "middle";

 ctx.font = "bold 25px Times, serif";

 ctx.fillText("Dogo", 0, 0);

 // Draw hash marks and numbers

 ctx.strokeStyle = "white";

 ctx.lineWidth = TICK\_WIDTH;

 ctx.textBaseline = "top";

 ctx.font = "20px Arial, sans-serif";

 ctx.beginPath();

 for (var i=0; i<NUM\_OF\_TICK\_MARKS; i++) {

 if (i % 5 == 0) {

 ctx.fillText(i.toString(), 0, -DIAL\_RADIUS + 22);

 ctx.moveTo(0, -DIAL\_RADIUS + 20);

 }

 else {

 ctx.moveTo(0, -DIAL\_RADIUS + 14);

 }

 ctx.lineTo(0, -DIAL\_RADIUS + 4);

 ctx.rotate(2 \* Math.PI / NUM\_OF\_TICK\_MARKS);

 } // end for

 ctx.stroke();

 // Restore canvas origin so next drawDial works.

 ctx.translate(-CENTER\_X, -CENTER\_Y);

} // end drawDial

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// This function rotates the dial counter-clockwise by one tick.

function nudgeLeft() {

 drawDial(-1);

 currentTick = (currentTick + 1) % NUM\_OF\_TICK\_MARKS; // EXTRA CREDIT

} // end nudgeLeft

// This function rotates the dial clockwise by one tick.

function nudgeRight() {

 drawDial(1);

 // EXTRA CREDIT:

 // The following modulus assignment should work according to the

 // mathematical definition of modulus, but it does not work because

 // JavaScript implements the modulus operator incorrectly.

 // currentTick = (currentTick - 1) % NUM\_OF\_TICK\_MARKS;

 currentTick--;

 if (currentTick < 0) {

 currentTick += NUM\_OF\_TICK\_MARKS;

 }

} // end nudgeRight

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// EXTRA CREDIT:

// This function rotates the dial counter-clockwise

// to the user-specified target number.

function leftToTarget(form) {

 var target; // tick number user wants to go to

 var moveTicks; // move this number of ticks on the dial

 target = parseInt(form.elements["target"].value);

 moveTicks = target - currentTick;

 if (moveTicks < 0) {

 moveTicks = NUM\_OF\_TICK\_MARKS + moveTicks;

 }

 // Every 1/10 of a second, move the dial one tick.

 for (var i=0; i<moveTicks; i++) {

 setTimeout(

 function () {

 drawDial(-1);

 }, i \* 100);

 } // end for

 currentTick = target;

} // end leftToTarget

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// This function rotates the dial clockwise

// to the user-specified target number.

function rightToTarget(form) {

 var target; // number user wants to go to

 var moveTicks; // move this number of ticks on the dial

 target = parseInt(form.elements["target"].value);

 moveTicks = currentTick - target;

 if (moveTicks < 0) {

 moveTicks = NUM\_OF\_TICK\_MARKS + moveTicks;

 }

 // Every 1/10 of a second, move the dial one tick.

 for (var i=0; i<moveTicks; i++) {

 setTimeout(

 function () {

 drawDial(1);

 }, i \* 100);

 } // end for

 currentTick = target;

} // end rightToTarget