10% - //HTML: code used inside the <body>

Choose any TEN options

☐ Contains one or more linked images: **<img src='http://...'>**
☐ Contains a button: **<button>**
☐ Contains a text input line: **<input>**
☐ Contains a text input area: **<textarea>**
☐ Contains a link to another document: **<a href=''>**
☐ Contains a header: **<h1> <h2> <h3> <h4> <h5> <h6>**
☐ Contains a paragraph: **<p>**
☐ Contains a line break: **
**
☐ Contains strong or emphasized text: ** **
☐ Contains a span or div: ** **
☐ Contains a list of items: **<li>**
☐ Contains a table: **<table>**
☐ Contains an HTML canvas: **<canvas>**
☐ Contains Scalable Vector Graphics: **<svg>**
☐ Contains a playable mp3 file: **<audio>**

10% - //CSS: code used inside <style>

Choose any TEN options

☐ Apply a style to a tag: **p { ... }**
☐ Apply a style to an ID: **#exampleID { ... }**
☐ Apply a style to a class: **.exampleClass { ... }**
☐ Sets the family of text: **font-family: ...**
☐ Sets the color of text: **color: ...**
☐ Sets the background color of text: **background-color: ...**
☐ Sets the size of text: **font-size: ...**
☐ Sets the weight of text: **font-weight: ...**
☐ Sets the style of text: **font-style: ...**
☐ Sets the width or height of an element: **width: | height: ...**
☐ Sets the positioning of an element: **position: ...**
☐ Positions an element using measurements: **top: | left: ...**
☐ Sets opacity: **opacity: ...**
☐ Sets CSS animation: **@keyframes ...**

☐ Contains other CSS as approved by instructor

5% - //INIT: code used inside the <script>

Choose any FIVE

☐ Uses numerical variables: **exampleNumber = 42;**
☐ Initializes a numerical variable as 0: **exampleNumber = 0;**
☐ Uses string variables: **exampleString = 'Hello!';**
☐ Initializes a string variable as: **exampleString = '';**
☐ Uses an array: **exampleArr = ['Hello', ' world!'];**
☐ Initializes an array as []: **exampleArr = [];**
☐ Uses an object: **exampleObj = { language: 'English', greeting: 'Hello!' };**
☐ Initializes an object as {}: **exampleObj = {};**
☐ Uses a two dimensional array: **exampleArr = [[1, 2, 3], [4, 5, 6]];**
☐ Uses a complex data structure (eg: an array of objects):

```
exampleArr = [
        { langauge: 'English', greeting: 'Hello!' },
        { langauge: 'German', greeting: 'Guten Tag!' }
];
```

☐ Uses three dimensional arrays
☐ Uses pointers (an ugly hack, but it is kind of possible)

☐ Contains other variable structures as approved by instructor

5% - //INPUT: trigger functions and read information

Choose any FIVE options

☐ Trigger things by clicking on an HTML element:
document.getElementById(...).onclick = exampleFunction;
☐ Trigger things (like the whole page) when they finish loading:
document.body.onload=exampleFunction()
☐ Trigger things by changing the contents of an input line or area:
document.getElementById(...).onchange=exampleFunction;
☐ Read information from an input line:
document.getElementById('inputID').value
☐ Read information from the keyboard:
window.onkeydown = function (keyEvent) { ... }
☐ Trigger a function by mousing over:
document.getElementById(...).onmouseover=exampleFunction;
☐ Trigger a function by moving the mouse:
document.onmousemove = function (mouseEvent) { ... }
☐ Trigger a function by pressing the mouse button:
document.onmousedown = function () { ... }
☐ Read the height or width of the window:
window.innerWidth | window.innerHeight
☐ Read the current position of an element:
document.getElementById('inputID').offsetTop | .offsetLeft
☐ Turning off the click or mouse functions of an element:
document.getElementById('exampleID').onmouseover = ''
☐ The program reads the time: **new Date()**
☐ The user uploads a file or a link
☐ Uses drop-down menus

☐ Contains other input as approved by instructor

15% - //PROCESS: code inside the <script>

Choose any FIFTEEN options

☐ ☐ Uses a singular if statement: **if (exampleVar == 1) { ... }**
☐ Uses an if...else if...else chain:

```
if ( exampleVar == 1 ) { ... }
else if ( exampleVar == 2 ) { ... }
else { ... }
```

☐ ☐ Uses a for loop: **for (i in exampleArr) { ... }**
☐ Uses a nested loop
☐ ☐ ☐ Uses string addition: **'Hello' + ' world!'**
☐ Uses a split function: **exampleVar.split(' ')**
☐ Uses a join function: **exampleArr.join(', ')**
☐ ☐ ☐ Uses arithmetic functions: **5 + 6 - 7 * 3 / 4**
☐ Uses rounding functions: **Math.round() | .floor() | .ceil()**
☐ Generates random numbers: **Math.random()**
☐ ☐ ☐ Uses additional math (powers, trig, etc.): **Math.pow() | Math.PI**
☐ Uses other string functions
☐ Uses regular expressions
☐ Uses other array functions
☐ Uses other object functions
☐ Uses time functions
☐ Uses animation callbacks:

```
$('#exampleID').animate(
        {top: '100px'},
        {duration: 1000,
         easing: 'linear',
         complete: function () { ... } }
);
```

☐ ☐ Uses setInterval/setTimeout
☐ ☐ Uses recursion or function callbacks
☐ Contains other process functions as approved by instructor

5% - //OUTPUT: code inside the <script>

Choose any FIVE options

☐ Outputs inside an html tag:
document.getElementById('exampleID').innerHTML
☐ Adds/removes/changes an image:
document.getElementById('exampleID').src
☐ Changes the position of an element:
document.getElementById('exampleID').style.top | .left
☐ Changes the style of an element:
document.getElementById('exampleID').style.backgroundColor
☐ Uses animation to change position:

```
$('#exampleID').animate(
        {top: '100px'},
        {duration: 1000, easing: 'linear'}
);
```

☐ Uses animation to change other style properties:

```
$('#exampleID').animate(
        {width: '100px'},
        {duration: 1000, easing: 'linear'}
);
```

☐ Creates a pop up window: **alert('Hi there!');**
☐ Sends data to the developer's console:
console.log('in mainProcedure...');
☐ Plays an audio clip:
document.getElementById('audioID').play();
☐ creates a new element:
document.createElement('img')
☐ appends an element to an ID:

```
document.getElementById( 'outputID' ).appendChild(
        newElement
);
```

☐ Uses other output as approved by instructor

20% - //Programming structure

Use ALL TEN options (each skill has double weight)

☐ ☐ Uses a function:
var exampleFunction = function () { ... }
☐ ☐ Use a function that accepts parameters/arguments as input:
var exampleFunction = function (exampleVariable) { ... }
☐ ☐ Uses a function that returns information:
return exampleVariable
☐ ☐ Uses an object-oriented structure with a property:
var exampleObject = { exampleProperty: 'example' }
☐ ☐ Uses an object-oriented structure with a method:
var exampleObject = { exampleMethod: function () {...} }
☐ ☐ Uses **.this** inside an object-oriented structure:
this.row = this.row + 1;
☐ ☐ Uses an object constructor:

```
var ExampleConstructor = function ( exampleVar ) {
        this.exampleProperty = exampleVar;
};
```

☐ ☐ Uses separate functions/methods for reading input
☐ ☐ Uses separate functions/methods for processing
☐ ☐ Uses separate functions/methods for outputting information

☐ ☐ Uses other program structure as approved by instructor

10% - //CHECKLIST

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ Completes and hands in an exam skills checklist

20% - //STYLE: Professional programming habits and communication

☐ The author of the program is indicated in the <head>
☐ Comment for the date started and dates modified, including what was modified
☐ Comment for describing the whole program
☐ Comment for describing each initialization block, function or method //INIT:
☐ Comment for describing each input function or method //INPUT:
☐ Comment for describing each processing function or method //PROCESS:
☐ Comment for describing each output function or method //OUTPUT:
☐ Descriptive comments for each function or method
☐ Descriptive comments for each loop
☐ Descriptive comments for each if statement

☐ Uses descriptive, full language variable names
☐ Uses descriptive, full language function and method names
☐ Uses camelCase for variable and function names
☐ Uses UpperFirst for object constructor names
☐ Avoids lines longer than 100 characters
☐ Creates sub-functions/methods if a function/method is over 20 lines
☐ Accurately indents code
☐ Orders program by: main → init → input → process → output
☐ Orders object code structures by: init → input → process → output
☐ Separates HTML clearly into head (meta/script/style) and body

Name: _____

Advice

- Get things working in bite-sized chunks.
- Use console logs to tell you where you are in your code.
- Use console logs to tell you what values your variables contain.
- Do CSS work when you are stuck or need a break.
- Plan your data structures carefully before you start.
- Have a clear definition of what you want input and what you want to output. This makes the process much easier.
- When looking online for advice, keep in mind that the quality of publicly available code is iffy. Get me to take a quick look at it...