FINAL ASSESSMENT RUBRIC

Guidelines for Grading

Below is an answer sheet for the final. Please note, when it comes to syntax/spelling/grammar, the answer should be marked wrong if the code will not compile or execute properly.

HTML / CSS

1. Create a CSS selector that selects all li tags nested in elements with a class of “list-items” that is nested in a ul element with a class of “list”. *[1pt]*

.list .list-items li

1. Taking the box model into account: If an element has a width of 400 pixels, padding of 15 pixels, a 4px border, and margin of 3 pixels. What is the total width of the element. *[1pt]*

444px

1. Create a CSS selector for an anchor element that will only apply its style when the user’s mouse passes over element. *[1pt]*

a:hover { }

1. Explain the differences between block, inline, and inline-block. *[1pt]*

Block elements are elements that take up the entire line(or width) of the page. Inline elements are elements that only take up the width required to render the element. Inline-block is an inline element but you can specify height and width.

JAVASCRIPT

1. Declare and initialize the variable temp to the number 50. Construct an if statement to check if the variable temp is above or equal to 60 (assume the temperature is in Farenheit). If true, the script should log the message, “You’ll be fine without a jacket.”. If false, the script should log the message, “It’s pretty cold out! Wear a jacket.” *[1pt]*

var temp = 50;

If (temp >= 60) { console.log(“You’ll be fine without a jacket.”) } else {

console.log(“It’s pretty cold out! Wear a jacket.”);

}

1. Create a for loop that logs the integers from 1 to 50 to the console. If a number is evenly divisible by 6, console.log the string “False” instead of the number. *[1pt]*

for (var i = 1; i <= 50; i++) { if (i % 6 === 0) { console.log(“False”); } }

1. Create a while loop that will prompt the user to enter their name until the user types “Macho Man”. *[1pt]*

var name = prompt(“Enter your name.”);

while(name !== “Macho Man”) { name = prompt(“Enter your name.”); }

1. Declare a function called findRemainder. This function should take two parameters and return parameter1 modulus parameter2. *[1pt]*

function findRemainder(param1, param2) { return param1 % param2; }

1. Declare a variable called movieNames, initialize it as an array of objects. Each object should have properties of name, year and director. Add an object for each of these catalog movies: *[1pt]*

|  |  |  |
| --- | --- | --- |
| Movie Name | Year Released | Director |
| “The Breakfast Club” | 1985 | “John Hughes” |
| “The Goonies” | 1985 | “Richard Donner” |
| “Dirty Dancing” | 1987 | “Emilie Ardolino” |

var movieNames = [ { name: “The Breakfast Club”, year: 1985, director: “John Hughes”}, { name: “The Goonies”, year: 1985, director: “Richard Donner”}, { name: “Dirty Dancing”, year: 1987, director: “Emilie Ardolino”} ];

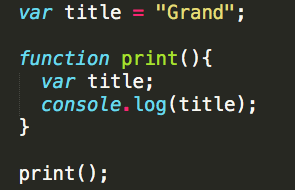
1. Using the array created above, use a forEach to log each movies name to the console. *[1pt]*

movieNames.forEach(function(movie) {

console.log(movie.name);

});

1. Consider the following code, what will be logged to the console? In a few sentences, explain why. *[1pt]*



It will print undefined. The variable title is being re-declared inside the print function with no value. The variable title, within the print function, is undefined.

JQUERY

1. Using jQuery, select an element with an id of main-link. Add an event handler using the on method. When the element is clicked, log the text of all paragraph tags to the console. *[1pt]*

$(“#main-link”).on(“click”, function() {

console.log($(“p”).text());

});

1. Using jQuery, make a GET request to the url “http://example.com/json”. Log the data from the request to the console. *[1pt]*

$.get(“<http://example.com/json>”).done(function(data) {

console.log(data);

});

$.get(“<http://example.com/json>”, function(data) {

console.log(data);

});

ANGULAR

1. What does MVC stand for? How does the cycle work? What are the parts? [1pt]

model view controller. The model is the information that is contained within the application. The view is what the user sees and interacts with. The controller is the business logic that processes information to update the model(which in turn, updates the view).

1. Using Angular, make a GET request to the url “http://api.example.com/movies” and assign the response to a variable called movies. *[1pts total]*

$http({

url: “http://api.example.com/movies”,

method: “GET”

}) .then(function(response) {

movies = response;

});

1. Create an angular component called movieDetails *[1pt]*
   1. Set the templateUrl property to “path/to/my/component.html”.
   2. Set the controller property to a function that logs “Movie Details” to the console

var movieDetails = {

templateUrl: “path/to/my/component.html”,

controller: function() {

console.log(“Movie Details”);

}

}

Write the HTML that would add this component to a view.

<movie-details></movie-details>

NODE.JS

1. Declare a variable http to require the http module. *[1pt]*

var http = require(“http”);

1. In a file called movie-checker.js is the following code…  
    var movies = [ "Breakfast Club", "Dirty Dancing", "The Goonies", "Jurassic Park", "8 Mile" ];  
    function checkMovies() {  
    movies.forEach(function(movie) {  
    if(movie === “8 Mile”) {

console.log(“Slim Shady is awesome”);

}

});  
 }  
 module.exports.movies = movies;  
 module.exports.checkMovies = checkMovies;  
  
Write the code required to import the content of movie-checker.js and call the checkMovies function. *[2pts total]*

var movieStuff = require(“./movie-checker.js”);

movieStuff.checkMovies();

1. What command is required to run a script file named script.js through NodeJS? *[1pt]*

node script.js