JavaScript

Answer Sheet

1. Including Imperative, what programming paradigms does JavaScript support?

Javascript is prototype oriented language, which means that JS hasn’t class concept, but inheritance performed by cloning existing instance of the object. It is also a multi-paradigm language, supporting imperative/procedural programming along with Object-Oriented Programming and functional programming.

1. What would be the output of the following code?

class Dog {

constructor(name) {

this.name = name;

}

speak() {

console.log(this.name + ' Barks!');

}

}

class Wolf extends Dog {

speak() {

super.speak();

console.log(this.name + ' Howls at the moon!');

}

}

var l = new Wolf('Fido');

l.speak();

.Output: Fido Barks!

Fido Howls at the moon!.

1. If given the following code what would the output be when the program executes the following line: console.log(myvar);

var myvar = 'my value';  
   
(function() {

var myvar;  
 console.log(myvar);  
 myvar = 'local value';  
})();

Output: Undefined.

1. What would the result be of the following function:

let myFunc = x => console.log(x+5);

myFunc(5);

Output: 10

1. What would the result be if you ran the following code:

function makeAdder(x) {  
 return function(y) {  
 return x + y;  
 };  
}  
var add8 = makeAdder(8);

var add12 = makeAdder(12);

console.log(add8(4));

console.log(add12(5));

Output: 12

17

1. What would the output of the following code be:

function sum() {  
 return this.a + this.b + this.c;  
}  
var o = {a: 3,b: 4,c: 5,

get average() {  
 return (this.a + this.b + this.c) / 3;  
 }  
};  
Object.defineProperty(o, 'sum', {get: sum, enumerable: true, configurable: true});  
console.log(o.average, o.sum);

Output: 4 12

1. Briefly, explain JavaScripts variable hoisting?

Hoisting is a JavaScript mechanism where variables and function declarations are moved to the top of their scope before code execution, which means  no matter where functions and variables are declared, they are moved to the top of their scope regardless of whether their scope is global or local.

1. What would the result be of the following function:

let myFunc = (x,y) => {

x++;

y++;

return x + y;

};

myFunc(10, 20);

Output: 32

1. What are the three different ways to do recursion in JavaScript?

Recursion is done via one of:

* 1. Call to function name
  2. Use of arguments.callee
  3. An in-scope variable referring to function

1. What makes a variable Global JavaScript? What makes a variable Local in JavaScript?

A variable declared outside a function will become as a global variable. A global variable has global scope for scripts and functions on a web page can access it. Local variables are created when a function starts, and deleted when the function is completed. Local variables have function scope, which means they can only be accessed from within the function. Since local variables are only recognized inside their functions, variables with the same name can be used in different functions.

Programming

Working individually define and test the functions described below.

**Your implementation of each function must have its function name spelled *exactly* as**

**it is in the description below.**

Your functions should output the results using the **console.log()** function**.** Your submission should consist of 1 .js file (Put all your functions into this one file). Your JavaScript source file should be named using your NetID. Example: cid021000.js. eLearning does not allow .js files to be uploaded. Zip the .js file with the word document containing the answers to the questions above. Upload the zip files to eLearning. You can find an online JavaScript interpreter at: <https://repl.it/languages>

EvenOdd

Write a recursive function that determines if a number is even or odd. Your function should take in one integer input and return if it is odd or even. You must use recursion. **Do NOT use the Modulus (%) operator.**

console.log(EvenOdd(5));

Odd

console.log(EvenOdd(10));

Even

console.log(EvenOdd(255));

Odd

console.log(EvenOdd("One"));

Not a valid number

console.log(EvenOdd(0));

Even

MyMathFunction

Write a function that takes in two numbers and a function name. The function should call the function name in the parameter passing it the two number to perform a math operation and returning the result. Your MyMathFunction should be able to handle 4 different math operations (add, subtract, divide, and multiply). You must write 5 functions to complete this assignment correctly (MyMathFunction, Add, Subtract, Multiply, Divide). Hint: Do not pass the function name as a string.

console.log(MyMathFunction(10, 10, Multiply));

100

console.log(MyMathFunction(50, 10, Divide));

5

console.log(MyMathFunction(10, 0, Divide));

Cannot Divide by 0!

console.log(MyMathFunction(20, 25, Add));

45

console.log(MyMathFunction(25, 20, Subtract));

5

SortMyArray

Write a function that takes in an array of integers and sorts the array from smallest to largest. Your function should return an array with the sorted list and print the list out. You can assume that the array being passed will always have integers (You do not need to type check the array). **DO NOT USE** the built-in array sort function.

var arr1=[-3,8,7,6,5,-4,3,2,1];

console.log(SortMyArray(arr1));

[ -4, -3, 1, 2, 3, 5, 6, 7, 8 ]

var test\_array=[1,2,3,4,5];

console.log(SortMyArray(test\_array));

[ 1,2,3,4,5]

var test\_array=[];

console.log(SortMyArray(test\_array));

Cannot sort an Empty Array!

var test\_array=[1,-1,-2,2,3,-3,-4,4,5,-5,0];

console.log(SortMyArray(test\_array));

[ -5,-4,-3,-2,-1,0,1,2,3,4,5]

var test\_array=[1, 1, 1, 100, 75, 2, 2, 37, 55, 55];

console.log(SortMyArray(test\_array));

[1,1,1,2,2,37,55,55,75,100]

myFib

Write a JavaScript function using recursion to compute the Fibonacci number of n (where n is a positive integer). Your function should output the calculated result for the n given. You also need to type check to make sure the value being given is an integer.

Fib(n) = 3 for n = 0 or n = 1

Fib(n) = Fib(n-1) + Fib(n-2) for n > 1

console.log(myFib(5));

24

console.log(myFib(10));

267

console.log(myFib(8));

102

console.log(myFib(“Thirtyfive”));

Not an integer value!

console.log(myFib(3.5));

Not an integer value!