Coding Questions

***Question: What is the value of foo?***

var foo = 10 + '20';

*Answer:* '1020', because of type coercion from Number to String

***Question: How would you make this work?***

add(2, 5); // 7

add(2)(5); // 7

*Answer:* A general solution for any number of parameters

'use strict';

let sum = (arr) => arr.reduce((a, b) => a + b);

let addGenerator = (numArgs, prevArgs) => {

return function () {

let totalArgs = prevArgs.concat(Array.from(arguments));

if (totalArgs.length === numArgs) {

return sum(totalArgs);

}

return addGenerator(numArgs, totalArgs);

};

};

let add = addGenerator(2, []);

add(2, 5); // 7

add(2)(5); // 7

add()(2, 5); // 7

add()(2)(5); // 7

add()()(2)(5); // 7

***Question: What value is returned from the following statement?***

"i'm a lasagna hog".split("").reverse().join("");

*Answer:* It's actually a reverse method for a string - 'goh angasal a m\'i'

***Question: What is the value of window.foo?***

( window.foo || ( window.foo = "bar" ) );

*Answer:* Always 'bar'

***Question: What is the outcome of the two alerts below?***

var foo = "Hello";

(function() {

var bar = " World";

alert(foo + bar);

})();

alert(foo + bar);

*Answer:*

* First: Hello World
* Second: Throws an exception, ReferenceError: bar is not defined

***Question: What is the value of foo.length?***

var foo = [];

foo.push(1);

foo.push(2);

*Answer:* .push is mutable - 2

***Question: What is the value of foo.x?***

var foo = {n: 1};

var bar = foo;

foo.x = foo = {n: 2};

*Answer:* undefined. Rather, bar.x is {n: 2}.

foo.x = foo = {n: 2} is the same as foo.x = (foo = {n: 2}). It is because a left term is first referenced and then a right term is evaluated when an assignment is performed in JavaScript. When foo.x is referenced, it refers to an original object, {n: 1}. So, when the result of the right term, {n: 2}, is evaluated, it will assigned to the original object, which is at the moment referenced by bar.

***Question: What does the following code print?***

console.log('one');

setTimeout(function() {

console.log('two');

}, 0);

console.log('three');

*Answer:* one, three and two. It's because console.log('two'); will be invoked in the next event loop.