ES6 Features - Exercises

# Exercise 01: Arrow Function

Given the following code, your task is to change function declaration/function expression of ES5 to Arrow Function syntax, however you have to make sure the output is still `**true**`

|  |
| --- |
| (function IIFE() {  function foo(x) {  var y = x \* 2;  return function bar(z) {  if (z.length > 3) {  return z.map(function baz(v) {  if (v > 3) return v + y;  else return baz(v \* 4);  });  } else {  var obj = [];  setTimeout(  function bam() {  obj.length = 1;  obj[0] = this.w;  }.bind(this),  100  );  return obj;  }  };  }  var p = foo(2);  var list1 = [1, 3, 4];  var list2 = list1.concat(6);  list1 = p.call({ w: 42 }, list1);  list2 = p(list2);  setTimeout(function() {  console.log(  list1[0] ===  list2.reduce(function(s, v) {  return s + v;  }, 0)  );  }, 200);  })(); |

# Exercise 02: Block Scope

Fix the following code, so the output is `**true**`

|  |
| --- |
| var x = 2,  fns = [];  (function() {  var x = 5;  for (var i = 0; i < x; i++) {  // ..  }  })();  // DO NOT MODIFY BELOW CODE  console.log(x \* 2 === fns[x \* 2]());  // true |

# Exercise 03: Rest/Spread

Use rest/spread operator so the code below display `**true**`

|  |
| --- |
| function foo() {}  function bar() {  var a1 = [2, 4];  var a2 = [6, 8, 10, 12];  return foo();  }  // DO NOT MODIFY BELOW CODE  console.log(bar().join('') === '281012');  // true |

# Exercise 04: Destructuring

Given the following code, you must use ES6 Destructuring feature to construct a data for function check so the output is `**true**`

|  |
| --- |
| function ajax(url, cb) {  // fake ajax response:  cb({  foo: 2,  baz: [6, 8, 10],  bam: {  qux: 12  }  });  }  function check(data) {  console.log(  56 ===  data.foo +  data.bar +  data.baz[0] +  data.baz[1] +  data.baz[2] +  data.bam.qux +  data.bam.qam  );  }  var defaults = {  foo: 0,  bar: 4,  bam: {  qux: 0,  qam: 14  }  };  // YOUR CODE HERE  function response(...) {  check(...); // true  }  // DO NOT MODIFY  ajax('http://fun.tld', response); |

# Exercise 05: Template String

Given the following code, you must complete the **upper** function and use it as a tag function for the template string `Hello \_\_\_\_ (@\_\_\_\_), welcome to the \_\_\_\_!` so the output is `**true`**

|  |
| --- |
| function upper(strings, ...values) {}  var name = 'Nguyen Van A',  account = 'ANV',  classname = 'FrontEnd AngularJS';  console.log(  `Hello \_\_\_\_ (@\_\_\_\_), welcome to the \_\_\_\_!` ===  'Hello NGUYEN VAN A (@ANV), welcome to the FRONTEND ANGULARJS!'  ); |