**Exercises**

*Lecture 2:*

ES6 - part 1

1. **Arrow functions**

1.1. Write an arrow function which returns the string ‘Hello world!’

1.2. Write an arrow function which returns the square of a given number

var x = 5;

var square = YOUR\_CODE\_HERE;

console.log(square(x);

1.3. Given this array: `[3,62,234,7,23,74,23,76,92]`, use the array filter method and an arrow function to create an array of the numbers greater than `70`

var numbers = [3, 62, 234, 7, 23, 74, 23, 76, 92];

var above70 = YOUR\_CODE\_HERE;

console.log(above70);

1.4. Write an expression using *filter* and *reduce* to compute the total value of the machines in the inventory array.

var inventory = [

{type: "machine", value: 5000},

{type: "machine", value: 650},

{type: "duck", value: 10},

{type: "furniture", value: 1200},

{type: "machine", value: 77}

];

var totalMachineValue = YOUR\_CODE\_HERE;

console.log(totalMachineValue);

1.5. Rewrite arrow function below with ES5 function

function Car() {

this.speed = 0;

setTimeout(() => {

this.speed += 10;

}, 100);

}

var c = new Car();

setTimeout(function() {

console.log(c.speed)

}, 200);

1. **Class**

Given an empty class named Animal.

class Animal {

//YOUR\_CODE\_HERE

}

Do the following tasks to completion this class body.

2.1. Implement **constructor** method for Animal class. The class has 3 properties: *name*, *thirst* and *belly*. The **constructor** method accepts 1 parameter which is *name.* Default values for *name* is value of *name* parameter, for *thirst* is number 100 and for *belly* is an empty array.

2.2. Implement ***drink()*** method which reduces value of *thirst* by 10 when being called.

2.3. Implement ***eat()*** method which accepts *food* parameter and pushes the value of *food* into *belly* array when being called.

2.4. Create a subclass of Animal named Dog which an additional property *breed* and method ***bark()***. Dog’s constructor accepts 2 parameter: *name* and *breed*. Default value for *breed* is value of *breed* parameter.

1. **Enhanced Object Literals**

3.1. Rewrite the function below using ES6 enhanced object literals. Write a getter (accessor) function for *price*

function getPrice(service) {

var price = 50.0;

return {

service: service,

\_price: price,

discount: function(rate) {

if (rate <= 0 || rate > 1) {

throw new Error('Invalid discount rate');

}

this.\_price \*= 1 - rate;

},

// YOUR\_GETTER\_FUNCTION\_HERE

};

}

var priceDetail = getPrice('EXPRESS');

priceDetail.discount(0.2);

console.log(priceDetail.price);

1. **Template string**
2. **Destructuring**

Given an object named person.

var person = {

name: 'Joe',

age: 21,

married: true,

hobbies: ['football', 'tennis', 'cooking'],

contact: {

phone: '0123456789',

email: 'name@example.com',

},

};

Using ES6 destructuring, do the following:

5.1. Get name and age of person object.

5.2. Create 2 variables myName and myAge which have value of name and age, respectively.

5.3. Get 2 first hobbies assigned to 2 variables (e.g. hobby1 & hobby2).

5.4. Write a function to get person’s married.

5.5. Write a function to get person’s phone and email. Default value of email is [you@mail.com](mailto:you@mail.com)