CodeAR S.A. – Software factory

**María**. Líder técnica a del área. En una persona muy ocupada y su rol consiste en traducir las necesidades del cliente a requerimientos técnicos.

**Juan**. Es pasante y está empezando en el mundo de la programación. Si bien es algo tímido, le gusta aprender cosas nuevas y encarar nuevos desafíos.

**Camila**. Estoy aprendiendo cosas como Juan, pero ya cuento con algunos conocimientos previos.

**Proyecto**: Concesionario de automóviles. Compra y venta de autos.

**Necesidad:** Construir una lista con todos los vehículos que tiene registrados.

ETAPA 0. Por cada vehículo necesito conocer la siguiente información:

Marca:

Modelo:

Color:

Año:

Kilómetros recorridos:

Precio:

Cuotas:

Patente:

Vendido:

let auto = {

marca : string,

modelo : string,

color : string,

anio : numérico,

kmRecorridos : numérico,

precio : numérico,

cuotas : numérico,

patente : string,

vendido : booleano

}

**ETAPA 1.** Requerir módulo autos. Crear un objeto literal llamado concesionaria que contendrá la lista de autos importada anteriormente.

let autos = require(“./autos”);

const concesionaria = {

autos: autos // no va entre “ ” porque no es un string lo que estoy llamando, es el nombre de la variable que declare arriba.

}

**ETAPA 2.** Crear la funcionalidad buscarAuto que reciba como parámetro la patente y devuelva el auto al cual le corresponde. En caso de no encontrar el mismo, deberá retornar null.

Recibe como parámetro una patente.

* si la encuentra en autos.patente 🡪 devuelve autos.marca
* si no la encuentra 🡪 devuelve null

buscarAuto: function(patente) {

let autos = **this**.autos // this.*nombreDeLoQueQuieroAcceder* - para acceder a la funcionalidad de una propiedad dentro del mismo objeto.

let autoPorPatente = null;

Funcion que recorra array (forEach), función con condición patente === auto.patente (if), si la condición se cumple autoPorPatente debería cambiar su valor “null” por el del auto encontrado (autoPorPatente = auto)

return autoPorPatente;

}

**ETAPA 3.** Agregar funcionalidad venderAuto que recibe la patente como parámetro y en caso de encontrarlo vacante le asigna el estado de vendido.

venderAuto: function(patente) {

hay que hacer uso de la función buscarAuto y si lo encuentra, cambiar el valor de objeto.vendido = true

}

**ETAPA 4.** Crear la funcionalidad autosParaLaVenta. Es decir, solo deben aparecer los autos con vendido:false.

autosParaLaVenta: function() {

let autos = this.autos

let autosDisponibles = autos.filter(function(auto){

return auto.vendido === false})

return autosDisponibles

}

**ETAPA 5.** Desarrollar funcionalidad autosNuevos. La condición es que auto.km < 100. Debemos requerir la función autosParaLaVenta.

autosNuevos: function() {

let autos0km = this.autosParaLaVenta()

return autos0km.filter(){ auto.km < 100 }

}

**ETAPA 6.** Completar la función listaDeVentas. Retorna una lista que contiene el precio de venta de cada auto vendido.

Devuelve un array de números (auto.precio) de los autos vendidos.

Es decir que si auto.vendido === true devuelvo auto.precio. Debería recorrer el array, preguntar en cada objeto por condición y que se sume a la lista el valor de auto.precio

listaDeVenta: function() {

let listaPrecioAutosVendidos = [] // declaro una variable con un array vacío que se irá completando a medida que itere en autos.

let autos = this.autos

autos.forEach(function(auto){ if(auto.vendido === true) listaPrecioAutosVendidos.push(auto.precio) })

return listaPrecioAutosVendidos

}

**ETAPA 7.** Función totalDeVentas. Devuelve la sumatoria del valor de todas las ventas realizadas. Hay que utilizar la función reduce().

totalDeVentas: function() {

let totalVentas = listaDeVenta()

totalVentas.reduce(function(acum, precio){ return acum + precio})

}

// Esto funcionaría si listaDeVenta no es un array vacío, es decir tiene valores dentro porque se vendieron autos. Pero, si no se vendió ningún auto y por lo tanto listaDeVentas está vacío, la ejecución de totalDeVentas me va a dar error.

Entonces debería usar un condicional para diferenciar a una listaDeVenta vacía de una con elementos:

If (listaDeVentas > 0) {return totalVentas.reduce()} else {return 0}

**ETAPA 8**. Desarrollar la función puedeComprar. Recibe por parámetro un auto y una persona. Devuelve true si la persona puede comprar el auto.

Dos factores para saber si una persona puede comprar el auto:

Costo total: auto.precio <= capacidadDePagoTotal

&&

Capacidad de pago en cuotas: auto.precio / auto.cuotas <= capacidadDePagoEnCuotas

Una persona va a ser representada mediante el siguiente objeto literal (lo declaro en un archivo como módulo y lo exporto para poder requerirlo luego):

Let persona = {  
nombre: “Juan”,  
capacidadDePagoEnCuotas: 20000,  
capacidadDePagoTotal: 100000  
}

module.exports = persona

puedeComprar: function(auto, persona) {

if ( auto.precio <= persona.capacidadDePagoTotal && auto.precio / auto.cuotas <= persona.capacidadDePagoEnCuotas ) { return true } else {return false}

}

**ETAPA 9.** Crear autosQuePuedeComprar. Recibe persona como parámetro y devuelve la lista de autos que puede comprar.

Entonces, la función debe:

* obtener función 🡪 autosParaLaVenta
* por cada uno de los autos debe probar si la persona puede comprarlo 🡪 es decir usar la condición de puedeComprar
* Retornar un array de objetos que tenga los autos que pueda comprar 🡪 Es decir a la función autosParaLaVenta aplicarle un .filter(fuction(auto){ con la condición de puedeComprar })

autosQuePuedeComprar: function(persona) {

let listaAutosVenta = this.autosParaLaVenta()

return listaAutosVenta.filter(fuction(auto){ return la condición de puedeComprar })

}

// me sirve solo la condición de puedeComprar para que me devuelva el objeto entero. Si uso la función entera, me devuelve true o false sin mostrarme los datos.