DESARROLLO WEB ENTORNO CLIENTE

UNIDAD 6

EJERCICIOS - OOP

**Contenido**

[Ejercicio 1. 3](#_Toc66014364)

[Ejercicio 2. 4](#_Toc66014365)

[Ejercicio 4. 5](#_Toc66014366)

[Ejercicio 5. 9](#_Toc66014367)

[Ejercicio 6. 12](#_Toc66014368)

[Ejercicio 7. 15](#_Toc66014369)

[Ejercicio 8. 16](#_Toc66014370)

[Ejercicio 10. 16](#_Toc66014371)

# Ejercicio 1.

Crear cuatro objetos **Coche**, con los atributos: **Marca**, **Modelo**, **Año**. Mostrar posteriormente todos los coches con sus datos.

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title>U6 - Ej1</title>

</head>

<body>

<div id="msg"></div>

<script>

var msg = document.getElementById("msg");

class Coche {

constructor(marca, modelo, anyo) {

this.marca = marca;

this.modelo = modelo;

this.anyo = anyo;

this.\_propietario = "";

}

get info\_coche() {

return (this.marca + " " + this.modelo + ". Año: " + this.anyo);

}

set anyo\_coche(anyo) {

this.anyo = anyo;

}

set propietario(nombre) {

this.\_propietario = nombre;

}

}

var coches = new Array;

var coche1 = new Coche("Ford", "Ka", "2012");

var coche2 = new Coche("Seat", "Ibiza", "2018");

alert(coche1.info\_coche);

alert(coche2.info\_coche);

coches.push(coche1);

coches.push(coche2);

/\*

for (i = 0; i < coches.length; i++) { // FOR para imprimir array

msg.insertAdjacentHTML("beforeend", "Entro en bucle. Vuelta: " + i);

msg.insertAdjacentHTML("beforeend", "coches[i]: " + Object.values(coches[i]));

}\*/

for (coche in coches) { // FOR IN equivalente

msg.insertAdjacentHTML("beforeend", "coches[i]: " + Object.values(coches[coche]));

}

</script>

</body>

</html>

# Ejercicio 2.

Crear una Clase con cuatro propiedades (Nombre de alumno y 3 notas). Ir almacenando los alumnos en un Array.

a) Permitir añadir y eliminar alumnos y notas.

b) Añadirle una nueva propiedad (nota media) mediante **prototype**

c) Mostrar el Array ordenado por nombres o por nota media

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title>U6 - Ej2</title>

</head>

<body>

<div id="msg"></div>

<script>

class Alumno {

constructor(nombre, nota1, nota2, nota3) {

this.nombre = nombre;

this.nota1 = nota1;

this.nota2 = nota2;

this.nota3 = nota3;

}

get info\_alumno(){

return "Nombre alumno: " + this.nombre +

". Nota 1eval: " + this.nota1 +

". Nota 2eval: " + this.nota2 +

". Nota 3eval: " + this.nota3;

}

get media\_alumno() {

var media = parseFloat(Math.round((this.nota1 + this.nota2 + this.nota3) / 3)).toFixed(2);

return media;

}

}

Alumno.prototype.notamedia = 0;

var msg = document.getElementById("msg");

var arrayAlumnos = new Array;

for (i = 0; i < 6; i++) {

var str = new String("Alumno\_" + (i + 1));

var n1 = Math.round((Math.random() \* 10));

var n2 = Math.round((Math.random() \* 10));

var n3 = Math.round((Math.random() \* 10));

var alumno = new Alumno(str, n1, n2, n3);

alumno.notamedia = alumno.media\_alumno;

arrayAlumnos.push(alumno);

}

arrayAlumnos.sort(function() { return Math.random() - 0.5 }); // DESORDENA

for (i = 0; i < arrayAlumnos.length; i++) {

msg.insertAdjacentHTML("beforeend", arrayAlumnos[i].info\_alumno + "<br/>");

msg.insertAdjacentHTML("beforeend", arrayAlumnos[i].notamedia + "<br/>")

}

msg.insertAdjacentHTML("beforeend", "<br/>");

arrayAlumnos.sort(function (a, b) { // ORDENA POR NOTA MEDIA

if (a.notamedia < b.notamedia)

return -1;

if (a.notamedia > b.notamedia)

return 1;

return 0;

});

for (i = 0; i < arrayAlumnos.length; i++) {

msg.insertAdjacentHTML("beforeend", arrayAlumnos[i].info\_alumno + "<br/>");

msg.insertAdjacentHTML("beforeend", arrayAlumnos[i].notamedia + "<br/>")

}

msg.insertAdjacentHTML("beforeend", "<br/>");

arrayAlumnos.sort(function (a, b) { // ORDENA POR NOMBRE

if (a.nombre < b.nombre)

return -1;

if (a.nombre > b.nombre)

return 1;

return 0;

});

for (i = 0; i < arrayAlumnos.length; i++) {

msg.insertAdjacentHTML("beforeend", arrayAlumnos[i].info\_alumno + "<br/>");

msg.insertAdjacentHTML("beforeend", arrayAlumnos[i].notamedia + "<br/>")

}

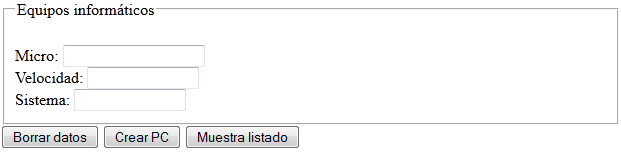
</script>

</body>

</html>

# Ejercicio 4.

Creación de clase PC con 3 propiedades (**micro**, **velocidad**, **sistema**) y 1 método (**verOrdenador**) que devuelva una cadena con las tres propiedades separadas por un ENTER.

* Se podrán crear infinitos elementos almacenándolos en un Array.
* Mostrar todos los ordenadores en una tabla dentro de una nueva ventana.
* <!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title>U6 OOP - Ej4</title>

<style>

\* {

margin-bottom: 5px;

}

table, td {

border: 1px solid black;

border-collapse: collapse;

}

label {

display: inline-block;

width: 80px;

}

tfoot td, thead td {

font-weight: bold;

}

</style>

</head>

<body>

<form id="formulario">

<fieldset>

<legend>Equipos informáticos</legend>

<label for="txt\_micro">Micro: </label><input type="text" id="txt\_micro" /><br />

<label for="txt\_velocidad">Velocidad: </label><input type="text" id="txt\_velocidad" /><br />

<label for="txt\_sistema">Sistema: </label><input type="text" id="txt\_sistema" />

</fieldset>

<div id="btns">

<input type="reset" value="Borrar datos" />

<input type="button" value="Crear PC" id="btn\_add" onclick="addPC()" />

<input type="button" value="Mostrar lista" id="btn\_show" onclick="showList()" />

</div>

</form>

<table id="tabla"></table>

<div id="msg"></div>

<script type="text/javascript">

// Elementos del html

var msg = document.getElementById("msg");

var tabla = document.getElementById("tabla");

var btns = document.getElementById("btns");

// Clase PC

class PC {

constructor(micro, velocidad, sistema) {

this.micro = micro;

this.velocidad = parseInt(velocidad);

this.sistema = sistema;

}

get verOdernador() {

var str = this.micro + ", " + this.velocidad + ", " + this.sistema + "<br/>";

return str;

}

}

function addPC() {

var nuevoPC = new PC(txt\_micro.value, txt\_velocidad.value, txt\_sistema.value);

lista\_pcs.push(nuevoPC);

formulario.reset();

}

function showList() {

// Borro posible contenido anterior

tabla.innerHTML = "";

// Creo tabla y encabezado

tabla.createCaption().innerHTML = "Lista de PCs";

var theader = tabla.createTHead(); // Creamos un <thead> vacío, enganchado a <table>

var rowH = theader.insertRow(0); // Creamos un <tr> vacío, enganchado al <thead>

//rowH.font-weight = "bold";

rowH.insertCell(0).innerHTML = "Num."; // Insertamos una nueva celda (<td>), enganchada al <tr>

rowH.insertCell(1).innerHTML = "Micro"; // Insertamos una nueva celda (<td>), enganchada al <tr>

rowH.insertCell(2).innerHTML = "Velocidad"; // Insertamos una nueva celda (<td>), enganchada al <tr>

rowH.insertCell(3).innerHTML = "Sistema"; // Insertamos una nueva celda (<td>), enganchada al <tr>

var tbody = tabla.createTBody();

for (i = 0; i < lista\_pcs.length; i++) {

var row = tbody.insertRow(i); // Creamos un <tr> vacío, enganchado al <table>

var obj = lista\_pcs[i];

console.log(obj);

var strObj = obj.verOdernador;

var arrObj = strObj.split(", ");

console.log(arrObj);

row.insertCell(0).innerHTML = (i+1);

for (j = 0; j < arrObj.length; j++) {

row.insertCell(j+1).innerHTML = arrObj[j];

}

if (i == (lista\_pcs.length - 1)) {

// Creo pie de tabla

var tfooter = tabla.createTFoot(); // Creamos un <tfoot> vacío, enganchado a <table>

var rowF = tfooter.insertRow(0);// Creamos un <tr> vacío, enganchado al <tfoot>

var cellF = rowF.insertCell(0); // Insertamos una nueva celda (<td>), enganchada al <tr>

cellF.colSpan = 3;

cellF.innerHTML = "Equipos listados:";

rowF.insertCell(1).innerHTML = (i + 1); // Insertamos una nueva celda (<td>), enganchada al <tr>

}

}

}

// Creo objetos

var pc1 = new PC("i7", 4770, "Windows");

var pc2 = new PC("i5", 3770, "Linux");

var lista\_pcs = new Array;

lista\_pcs.push(pc1);

lista\_pcs.push(pc2);

console.log(Object.keys(pc1));

console.log(Object.values(pc1));

</script>

</body>

</html>

# Ejercicio 5.

Crear un objeto Persona que tenga dos propiedades: **Nombre** y **DineroAhorrado**, y dos métodos: **Ahorra** y **Gasta**. Crea un Array con varias personas (con distintos nombres y distintas cantidades de dinero ahorrado). Y muestra una pantalla similar a la siguiente, desde la que se pueda seleccionar una persona (bien con el nombre o con su posición en la tabla) y una cantidad que podrá ahorrar o gastar. Lógicamente, habrá que comprobar que no pueda gastar más dinero del que tenga ahorrado.

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<style>

table, td {

border: 1px solid black;

text-align: center;

}

</style>

<title>U6 Ejer5</title>

</head>

<body>

<form id="formulario">

<table>

<tr>

<td><label for="form\_persona">Persona: </label></td>

<td><input id="form\_persona" type="text" /></td>

</tr>

<tr>

<td><label for="form\_cantidad">Cantidad: </label></td>

<td><input id="form\_cantidad" type="text" /></td>

</tr>

<tr>

<td colspan="2">

<input type="button" id="btn\_ahorrar" value="Ahorrar" onclick="ahorrar(form\_persona.value,form\_cantidad.value)" />

<input type="button" id="btn\_gastar" value="Gastar" onclick="gastar(form\_persona.value,form\_cantidad.value)" />

</td>

</tr>

<tr>

<td colspan="2"><input type="button" id="btn\_lista" value="Listas Grupo" onclick="listar()" /></td>

</tr>

</table>

</form>

<div id="msg"></div>

<script type="text/javascript">

class Persona {

constructor(nombre, ahorros) {

this.nombre = nombre;

this.ahorros = parseInt(ahorros);

}

get info\_persona() {

return this.nombre + ", " + this.ahorros + "<br/>";

}

set ahorra(cuanto) {

this.ahorros += parseInt(cuanto);

}

set gasta(cuanto) {

this.ahorros -= parseInt(cuanto);

}

}

function ahorrar(quien, cuanto) {

var i;

if (isNaN(quien)) { // Compruebo si pasan el nombre o el índice

i = busca(quien);

} else {

i = quien;

}

if (i > -1 && i < personas.length) { // Compruebo que sea un índice válido

personas[i].ahorra = parseInt(cuanto);

msg.insertAdjacentHTML("beforeend", personas[i].info\_persona);

} else {

msg.insertAdjacentHTML("beforeend", "NO ENCONTRADO<br/>");

}

}

function gastar(quien, cuanto) {

var i;

if (isNaN(quien)) { // Compruebo si pasan el nombre o el índice

i = busca(quien);

} else {

i = quien;

}

if (i > -1 && i < personas.length) { // Compruebo que sea un índice válido

personas[i].gasta = parseInt(cuanto);

msg.insertAdjacentHTML("beforeend", personas[i].info\_persona);

} else {

msg.insertAdjacentHTML("beforeend", "NO ENCONTRADO<br/>");

}

}

function listar() {

msg.innerHTML = "";

for (i = 0; i < personas.length; i++) {

msg.insertAdjacentHTML("beforeend", Object.values(personas[i]) + "<br/>");

}

}

function busca(obj\_nombre) {

for (i = 0; i < personas.length; i++) {

var arrPropiedades = Object.values(personas[i]); // Guardo en un array values

if (arrPropiedades.indexOf(obj\_nombre) != -1) { // Si lo encuentra

return arrPropiedades.indexOf(obj\_nombre); // Devuelvo index

}

}

return -1; // No encontrado

}

var person1 = new Persona("Roman Fuster", parseInt(27000));

var person2 = new Persona("Tuko Tukeras", parseInt(90000));

person1.ahorra = 1000;

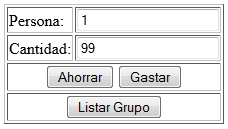
var personas = new Array;

personas.push(person1);

personas.push(person2);

</script>

</body>

</html>

# Ejercicio 6.

Realizar una clase que permita almacenar:

Usuario

Clave de acceso

Pregunta de seguridad

Respuesta

Permitir Altas, Bajas, Modificaciones, Consultas individuales, Listados

<!DOCTYPE html>

<html lang="es" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title>U6 Ej6</title>

</head>

<body>

<form id="acciones">

<fieldset style="border:none;">

<legend>Seleccione una opción</legend>

<input type="button" id="btn\_listar" value="Listar Usuarios" onclick="listar()" />

<input type="button" id="btn\_alta" value="Alta usuario" onclick="alta()" />

<input type="button" id="btn\_baja" value="Baja usuario" onclick="baja()" />

<input type="button" id="btn\_modifica" value="Modificar clave" onclick="modifica()" />

</fieldset>

</form>

<form id="formulario" style="display: none;">

<fieldset>

<legend id="leyenda"></legend>

<div id="login">

<label for="txt\_nombre">Nombre: </label><input type="text" id="txt\_nombre" class="entrada" />

<label for="txt\_clave">Contraseña: </label><input type="text" id="txt\_clave" class="entrada" />

</div>

<div id="seguridad" style="display: none;">

<label for="txt\_pregunta">Pregunta de seguridad: </label><input type="text" id="txt\_pregunta" class="entrada" />

<label for="txt\_respuesta">Respuesta: </label><input type="text" id="txt\_respuesta" class="entrada" />

</div>

<div id="botones">

<input type="reset" id="btn\_borra" value="Borrar" />

<input type="button" id="btn\_envia" value="Confirmar" onclick="" autofocus/>

</div>

</fieldset>

</form>

<div id="listado"></div>

<div id="msg"></div>

<script type="text/javascript">

// Elementos HTML

var lista = document.getElementById("lista");

var leyenda = document.getElementById("leyenda");

var msg = document.getElementById("msg");

class Usuario {

constructor(nombre, clave, pregunta, respuesta) {

this.nombre = nombre;

this.clave = clave;

this.pregunta = pregunta;

this.respuesta = respuesta;

}

get info\_usuario() {

return this.nombre + ", " + this.clave + ", " + this.pregunta + ", " + this.respuesta + "<br/>";

}

get get\_clave() { return this.clave; }

set set\_clave(nueva\_clave) { this.clave = nueva\_clave; }

set set\_pregunta(nueva\_pregunta) { this.pregunta = nueva\_pregunta; }

set set\_respuesta(nueva\_respuesta) { this.respuesta = nueva\_respuesta; }

}

function comprueba() {

var entradas = document.getElementsByClassName("entrada");

for (i = 0; i < entradas.length; i++) {

if (entradas[i].type == "text" && entradas[i].value == "") {

return false;

}

}

return true;

}

function listar() {

listado.innerHTML = "<h2>Lista de usuarios:</h2>";

for (i = 0; i < users.length; i++) {

listado.insertAdjacentHTML("beforeend", users[i].info\_usuario);

}

}

function alta() {

leyenda.innerHTML = "Alta de nuevo usuario";

formulario.style = "display: display;"

seguridad.style = "display: display;"

btn\_envia.onclick = function () {

if (comprueba()) {

var nuevo\_user = new Usuario(txt\_nombre.value, txt\_clave.value, txt\_pregunta.value, txt\_respuesta.value);

users.push(nuevo\_user);

msg.innerHTML = "";

listar();

} else {

msg.innerHTML = "COMPLETE TODOS LOS CAMPOS";

}

}

}

function baja() {

leyenda.innerHTML = "Baja de usuario";

formulario.style = "display: display;"

seguridad.style = "display: none;"

btn\_envia.onclick = function () {

var i = busca(txt\_nombre.value);

if ((i != -1) && (busca(txt\_clave.value) == i)) { // Si se encuentra el usuario y la clave

//console.log(true);

users.splice(i, 1); // Eliminar usuario

msg.innerHTML = "";

listar();

} else {

msg.innerHTML = "No coincide";

}

}

}

function modifica() {

leyenda.innerHTML = "Modifica usuario";

formulario.style = "display: display;"

seguridad.style = "display: none;"

btn\_envia.onclick = function () {

var i = busca(txt\_nombre.value);

if ((i != -1) && (busca(txt\_clave.value) == i)) { // Si se encuentra el usuario y la clave

//console.log(true);

seguridad.style = "display: display;"

txt\_pregunta.value = users[i].pregunta;

txt\_pregunta.disabled = "disabled";

msg.innerHTML = "";

btn\_envia.onclick = function () {

var nuevaClave = prompt("Introduzca nueva clave");

users[i].set\_clave = nuevaClave;

listar();

}

} else {

msg.innerHTML = "No coincide";

}

}

}

function busca(obj\_valor) {

console.log(obj\_valor);

for (i = 0; i < users.length; i++) {

var arrPropiedades = Object.values(users[i]); // Guardo en un array values

if (arrPropiedades.indexOf(obj\_valor) != -1) { // Si lo encuentra

return i; // Devuelvo index array users

}

}

return -1; // No encontrado

}

var user1 = new Usuario("Vero", "B3r0", "Perrin?", "Tuko");

var user2 = new Usuario("Roman", "r0m4n", "Año nac", "1983");

var users = new Array;

users.push(user1);

users.push(user2);

</script>

</body>

</html>

# Ejercicio 7.

Añadir un método **trim()** a la Clase **String**, que permita quitar los espacios en blanco de una cadena, de cualquier posición.

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title></title>

</head>

<body>

<script type="text/javascript">

String.prototype.trimAll = function () {

return this.replace(/\s+/g, '');

};

/\*if (!String.prototype.trim) {

(function () {

String.prototype.trim = function () {

return this.replace(/\s+/g, '');

};

})();

}\*/

texto = " Hola y adios ";

console.log(texto);

console.log(texto.length);

texto\_2 = texto.trimAll();

console.log(texto\_2);

console.log(texto\_2.length);

</script>

</body>

</html>

# Ejercicio 8.

Crear un método de la clase String que trunque una cadena a la longitud dada, y añada un indicador de cadena truncada.

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title>U6 Ej8</title>

</head>

<body>

<script type="text/javascript">

String.prototype.recorta = function (num) {

var str = this.slice(0, num);

str += "(...)";

return str;

};

var texto = "En un lugar de la mancha de cuyo nombre no quiero acordarme";

var indice = 15;

var texto\_2 = texto.recorta(indice);

console.log(texto);

console.log(texto\_2);

</script>

</body>

</html>

# Ejercicio 10.

Haz una clase llamada **Persona** que siga las siguientes condiciones:

* Sus Propiedades son: **nombre, edad, DNI, sexo**(H hombre, M mujer)**, peso y altura**. Si quieres añadir alguna otra propiedad, puedes hacerlo. Impedir que puedan meter datos erróneos.
* Los métodos que se implementaran son:
  + **calcularIMC()**: calculara si la persona está en su peso ideal (peso en kg/(altura^2  en m)), si esta fórmula devuelve un valor menor que 20, la función devuelve un -1, si devuelve un número entre 20 y 25 (incluidos), significa que está por debajo de su peso ideal la función devuelve un 0  y si devuelve un valor mayor que 25 significa que tiene sobrepeso, la función devuelve un 1. Te recomiendo que uses constantes para devolver estos valores.
  + **esMayorDeEdad()**: indica si es mayor de edad, devuelve un booleano.
  + **toString()**: devuelve toda la información del objeto.

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title>U6 Ej10</title>

</head>

<body>

<div id="msg"></div>

<script>

class Persona {

constructor(nombre, edad, dni, sexo, peso, altura) {

this.nombre = nombre;

this.edad = parseInt(edad);

this.dni = dni;

this.sexo = sexo;

this.peso = parseFloat(peso);

this.altura = parseInt(altura);

}

get toString() {

return this.nombre + " de " + this.edad + " años y dni " + this.dni + ", pesa " + this.peso + "Kg y mide " + this.altura + "cm. <br/>";

}

get esmayordeedad() {

if (this.edad > 17) {

return true;

} else {

return false;

}

}

get calcularIMC() {

var calculoIMC = this.peso / Math.pow((this.altura / 100), 2);

return Math.round(calculoIMC\*100)/100;

}

}

var msg = document.getElementById("msg");

var person1 = new Persona("Vero", 37, "71885796d", "M", 51, 165);

msg.insertAdjacentHTML("beforeend", person1.toString);

msg.insertAdjacentHTML("beforeend", "¿Es mayor de edad?: " + person1.esmayordeedad);

msg.insertAdjacentHTML("beforeend", "<br/>Cálculo IMC: " + person1.calcularIMC);

console.log(person1);

</script>

</body>

</html>

## Ejercicio 12.

Crear una clase Libro que contenga las siguientes propiedades:

* ISBN
* Titulo
* Autor
* Número de páginas

Crear el método **toString()** para mostrar la información relativa al libro con el siguiente formato:

“El libro con ISBN creado por el autor tiene páginas”

Además de poder dar de alta, dar de baja y modificar cualquier libro, permitir conocer cuál de todos los libros, tiene más páginas.

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title></title>

</head>

<body>

<div id="msg"></div>

<script>

class Libro {

constructor(isbn, titulo, autor, paginas) {

this.isbn = isbn;

this.titulo = titulo;

this.autor = autor;

this.paginas = paginas;

this.setIsbn = function (x) { this.isbn = x };

this.setTitulo = function (x) { this.titulo = x };

this.setAutor = function (x) { this.autor = x };

this.setPaginas = function (x) { this.paginas = x };

}

get toString() {

return "El libro titulado '" + this.titulo + "' con ISBN: " + this.isbn + " creado por el autor " + this.autor + " tiene " + this.paginas + " páginas";

}

}

function creaLibro(isbn, titulo, autor, paginas) {

var nuevoLibro = new Libro(isbn, titulo, autor, paginas);

vector.push(nuevoLibro);

}

function borraLibro(isbn\_libro) {

for (i = 0; i < vector.length; i++) {

if (vector[i].isbn === isbn\_libro) {

vector.splice(i, 1);

msg.insertAdjacentHTML("beforeend", "<p style = 'color: green;'>Libro borrado</p>");

break;

}

if (i == vector.length - 1) {

msg.insertAdjacentHTML("beforeend", "<p style='color: red;'>No existe ese libro, no se puede borrar</p>");

}

}

}

function mostrarLibros(array) {

msg.insertAdjacentHTML("beforeend", "<p>Mostrar libros:</p>");

for (i = 0; i < array.length; i++) {

msg.insertAdjacentHTML("beforeend", array[i].toString + "<br/>");

}

}

function masLargo() {

var numpags = 0;

var libroLargo = null;

for (objeto in vector) {

if (vector[objeto].paginas > numpags) {

libroLargo = vector[objeto];

numpags = vector[objeto].paginas;

}

}

msg.insertAdjacentHTML("beforeend", "El libro más largo es: " + libroLargo.toString + "<br/>");

}

/\*! Ordenar elementos del arreglo por una propiedad numérica. https://deployando.net/2012/08/16/js-ordenar-arreglos-objetos/ \*/

// No sé porqué hace que no funcione bien mostrarLibros() si uso bucle for in

Array.prototype.orderByNumber = function (property, sortOrder) {

// Primero se verifica que la propiedad sortOrder tenga un dato válido.

if (sortOrder != -1 && sortOrder != 1) sortOrder = 1;

this.sort(function (a, b) {

// La función de ordenamiento devuelve la comparación entre property de a y b.

// El resultado será afectado por sortOrder.

return (a[property] - b[property]) \* sortOrder;

})

}

var msg = document.getElementById("msg");

var vector = new Array();

msg.innerHTML = "<h2>Listado de libros</h2>";

creaLibro(12345, "Hola", "Pepe Palotes", 142);

creaLibro(456123, "Adios", "Ana Rodriguez", 222);

creaLibro(74523, "Otro libro", "Asunción Fdez", 500);

creaLibro(3657812, "Titulo desconocido", "Paco Pil", 167);

mostrarLibros(vector);

borraLibro(123450); // prueba para que falle

borraLibro(12345); // borrado

mostrarLibros(vector); // vuelvo a mostrar array

creaLibro(12345, "Hola", "Pepe Palotes", 142); // vuelvo a crear libro borrado

masLargo(); // devuelvo el más largo

vector[0].setTitulo("Nuevo titulo"); // modifico un libro

vector[0].setIsbn(123);

vector[0].setAutor("Mariano Perez");

vector[0].setPaginas(115);

msg.insertAdjacentHTML("beforeend", vector[0].toString); // lo muestro

// EXTRA: Ordenar array por número de páginas

vectorOrdenado = Array.from(vector); // clona array

vectorOrdenado.orderByNumber("paginas", 1); // de menor a mayor

msg.insertAdjacentHTML("beforeend", "<h3>Vector tal cual: </h3>");

mostrarLibros(vector);

msg.insertAdjacentHTML("beforeend", "<h3>Vector ordenado: </h3>");

mostrarLibros(vectorOrdenado);

</script>

</body>

</html>

## Ejercicio 13.

Nos piden hacer un programa orientado a objetos sobre un cine (solo de una sala) tiene un conjunto de asientos (8 filas por 9 columnas, por ejemplo).

Del cine nos interesa conocer la película que se está reproduciendo y el precio de la entrada en el cine.

De las películas nos interesa saber el título, duración, edad mínima y director.

Del espectador, nos interesa saber su nombre, edad y el dinero que tiene.

Los asientos son etiquetados por una letra (columna) y un número (fila), la fila 1 empieza al final de la matriz como se muestra en la tabla. También deberemos saber si está ocupado o no el asiento.

8A 8B 8C 8D 8E 8F 8G 8H 8I  
7A 7B 7C 7D 7E 7F 7G 7H 7I  
6A 6B 6C 6D 6E 6F 6G 6H 6I  
5A 5B 5C 5D 5E 5F 5G 5H 5I  
4A 4B 4C 4D 4E 4F 4G 4H 4I  
3A 3B 3C 3D 3E 3F 3G 3H 3I  
2A 2B 2C 2D 2E 2F 2G 2H 2I  
1A 1B 1C 1D 1E 1F 1G 1H 1I

Realizaremos una pequeña simulación, en el que generaremos muchos espectadores y los sentaremos aleatoriamente (no podemos donde ya este ocupado).

Solo se podrá sentar si tienen el suficiente dinero, hay espacio libre y tiene edad para ver la película, en caso de que el asiento este ocupado le buscamos uno libre.

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title>Cine</title>

</head>

<body>

<div id="msg"></div>

<script>

var id = 0;

class Sala {

constructor(columnas, filas, pelicula, precio) {

this.nombre = "Sala" + id++;

this.asientos = creaAsientos(columnas, filas);

this.pelicula = pelicula;

this.precio = precio;

}

get toString() {

return "<p><b>" + this.nombre + "</b>. Nº asientos: " + (this.asientos.length \* this.asientos[0].length) + ". Pelicula actual: " + this.pelicula.titulo + ". Precio entrada: " + this.precio + " euros</p>";

}

}

function creaAsientos(columnas, filas) {

var asientos = new Array();

for (let i = 0; i < filas; i++) {

//var col = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'; // Forma poco elegante

var char = 65;

var str = new Array();

for (let x = 0; x < columnas; x++) {

//str.push(col[x] + (i + 1));

str.push(String.fromCharCode(char) + (i + 1));

char++;

}

asientos.push(str);

}

asientos.reverse();

console.log(asientos);

return asientos;

}

class Pelicula {

constructor(titulo, duracion, edadMinima, director) {

this.titulo = titulo;

this.duracion = duracion;

this.edadMinima = edadMinima;

this.director = director;

}

get toString() {

return "<p><b>" + this.titulo + "</b>. Duración: " + this.duracion + " min. Edad mínima: " + this.edadMinima + ". Director: " + this.director + "</p>";

}

}

class Espectador {

constructor(nombre, edad, dinero) {

this.nombre = nombre;

this.edad = edad;

this.dinero = dinero;

this.asiento;

this.setAsiento = function (x) { this.asiento = x };

}

get getDinero() {

return this.dinero;

}

get toString() {

return "<p>" + this.nombre + ", " + this.edad + " años, " + this.dinero + " euros. Asiento: " + ((this.asiento == null) ? "Sin asignar" : this.asiento) + "</p>";

}

}

function creaEspectadores(n) {

for (let i = 0; i < n; i++) {

//var nombre = "nombre" + (i + 1);

var nombre = randomLetra() + randomLetra() + randomLetra();

var edad = Math.floor((Math.random() \* 40)) + 1; // Pruebo desde 1 a 40 años para acotar

var dinero = Math.floor((Math.random() \* 50)); // Pruebo de 0 a 50 euros

var persona = new Espectador(nombre, edad, dinero);

arrayEspectadores.push(persona);

}

}

function asignaAsientos(sala) {

var asientosLibres = Array.from(sala.asientos); // clono array de asientos

var numEspectadores = arrayEspectadores.length;

try {

for (let i = 0; i < arrayEspectadores.length; i++) { // recorro array

//console.log(asientosLibres);

do {

var randomF = Math.floor((Math.random() \* (sala.asientos.length)));

var randomC = Math.floor((Math.random() \* (sala.asientos[randomF].length)));

} while (asientosLibres[randomF][randomC] == null);

if (arrayEspectadores[i].dinero >= sala.precio) {

console.log(arrayEspectadores[i].edad);

console.log(sala.pelicula.edadMinima);

if (arrayEspectadores[i].edad >= sala.pelicula.edadMinima) {

arrayEspectadores[i].setAsiento(asientosLibres[randomF][randomC]); // asigno asiento al espectador

asientosLibres[randomF].splice(randomC, 1); // borro asiento del array de libres

console.log(asientosLibres);

console.log(arrayEspectadores[i]);

} else {

msg.insertAdjacentHTML("beforeend", "<p style='color: red;'>El espectador " + arrayEspectadores[i].nombre + " no tiene edad para ver " + sala.pelicula.titulo + "</p>");

}

} else {

msg.insertAdjacentHTML("beforeend", "<p style='color: red;'>El espectador " + arrayEspectadores[i].nombre + " no tiene suficiente dinero para la entrada de " + sala.pelicula.titulo + "</p>");

}

}

} catch (error) {

console.error(error);

}

}

function randomLetra() {

var char = Math.floor((Math.random() \* (90 - 65))) + 65;

return String.fromCharCode(char);

}

function randomPeli() {

var random = Math.floor((Math.random() \* (arrayPeliculas.length)));

return arrayPeliculas[random];

}

function muestra(array) {

var texto = array[0].constructor.name;

(texto == "Espectador") ? texto += "es" : texto += "s";

msg.insertAdjacentHTML("beforeend", "<h3>" + texto +"</h3>");

for (let i = 0; i < array.length; i++) {

console.log(array[i].toString);

texto = array[i].toString;

msg.insertAdjacentHTML("beforeend", texto);

}

}

var msg = document.getElementById("msg");

var arrayPeliculas = new Array();

arrayPeliculas.push(new Pelicula("Pelicula 1", 120, 13, "Director 1"));

arrayPeliculas.push(new Pelicula("Pelicula 2", 162, 18, "Director 2"));

arrayPeliculas.push(new Pelicula("Pelicula 3", 95, 0, "Director 3"));

arrayPeliculas.push(new Pelicula("Pelicula 4", 180, 7, "Director 4"));

var arraySalas = new Array();

arraySalas.push(new Sala(4, 5, randomPeli(), 13));

arraySalas.push(new Sala(3, 3, randomPeli(), 12));

var arrayEspectadores = new Array();

muestra(arraySalas);

muestra(arrayPeliculas);

creaEspectadores(15);

muestra(arrayEspectadores);

asignaAsientos(arraySalas[0]);

muestra(arrayEspectadores);

</script>

</body>

</html>