**Coderhouse - Camada 1662**

Lista de trabajos prácticos a realizar:

**TP1: Realizar una calculadora que realice las siguientes operaciones sumar, restar, multiplicar y dividir:**

* Deberá ingresar un valor mediante un prompt
* Deberá ingresar otro valor mediante otro prompt
* Mostrar el resultado de cada operación mediante distintos alert

**TP2: Días de la semana**

* Deberá poder ingresar un día de la semana
* Deberá poder compararlo con todos los días de la semana
* Deberá evaluar si es un día de semana y fin de semana mediante un mensaje similar a 'El día es hábil' sino 'El día es fin de semana';
* En caso de encontrar el día utilizar la sentencia break y terminar el for

# TP3: Mejorar la calculadora separando en funciones cada operación

* Deberá poder ingresar 2 valores
* Deberá poder ingresar que operación quiere realizar
* Cada operación tendrá que ser una función con los 2 paramentos para sumar, restar, multiplicar o dividir;
* Deberá mostrar el resultado de la operación en consola

**TP4: Alumnos**

Ingresar un listado de nombres de alumnos por pantalla. Cada vez que se ingresa un nuevo usuario, preguntar si se quiere ingresar otro utilizando la función "Confirm". En caso de que el usuario no quiera ingresar otro alumno, mostrar los siguientes resultados por consola:

* Indicar la cantidad de alumnos que se ingresaron
* Preguntar si "Maradona" vino a la clase. Hacerlo con otra función que tome un array como parámetro y retorne true o false dependiendo si lo encontró
* Mostrar el listado de alumnos por pantalla.

**TP5: Películas**

* Crear un array de películas
* Crear una función que agregue objetos “película” al array “películas”: ​El objeto película deberá tener ID, titulo, descripción, año, duración, actores, director. Deberá tener métodos para editar todas sus propiedades menos su ID, y para los actores tendrá que ir agregando uno a uno

**TP6: Crear el módulo IMDB**

* El array de películas tendrá que ser privado del modulo
* El modulo tendrá que poder agregar una película (ID y título)
* Validar que la película ingresada no se encuentre y en caso de encontrarla mostrar un mensaje advirtiendo el error
* El módulo tendrá que eliminar una película por ID
* El módulo tendrá que persistir el array de películas en session o local storage para que luego se pueda recuperar y seguir agregando o eliminando

**TP7: Mejorar el módulo IMDB**

* **​**El módulo deberá, de forma privada o pública, mostrar/renderizar las películas en el DOM
* El objeto película ahora deberá tener la propiedad descripción e imagen
* El título deberá ser un H1, la descripción un P y la imagen un img
* Mejorar el módulo IMDB

**TP8: Mejorar el módulo IMDB (Parte 2)**

* ​Crear un formulario que tenga los campos título, descripción y uno donde poder ingresar un link a la imagen.
* Este formulario tendrá que tener un botón ("agregar"), el cual deberá permitir al módulo IMDB agregar una película.
* Este formulario deberá tener también un botón ("mostrar todos"), el cual deberá permitir al módulo IMDB mostrar todas las películas
* Incluir otros botones para poder ordenar por AZ ZA e ID y mostrarlos
* En caso de encontrarse películas persistidas tendrán que volver a renderizarse
* Mejorar el módulo IMDB

**TP9 - Realizar un ToDo List, deberá tener las siguientes funcionalidades:**

* ​Crear el módulo ToDoList
* El módulo deberá tener los métodos necesarios para Agregar, Editar, Mostrar Todos, Buscar y Eliminar.
* Una tarea contiene las siguientes propiedades: id, titulo, descripción y completado
* Se tiene que poder editar título y descripción de una tarea
* Cambiar a estado completado y descompletado(?) una tarea
* Eliminar una tarea o todas las tareas
* Ordenar por A-Z y Z-A las tareas ya sea por título como por ID
* Las tareas deberán quedar persistidas mediante localstorage y poder mostrarse en caso de recargar el browser

**TP10: Lista de Artículos**

Crear un formulario con 3 campos:

* Título del Artículo.
* Texto del Artículo
* importancia (será un select box con los valores: “normal”, “novedad” y “urgente” )

A partir de un botón que diga “agregar” ir agregando las noticias en un contenedor en la página. Dependiendo del valor que tenga el select “importancia”, se cambiaran los estilos de cada noticia. Para eso, deberán generarse previamente clases de CSS para cada uno de los valores obtenidos del select.

**TP11: Curriculum Vitae.**

* Crear un formulario con los siguientes campos
  + Nombre (input=text)
  + Apellido (input=text)
  + Género (2 radio button: )
  + Edad (input=number)
  + Ocupación (select)
  + Estudios -> (input=text)
  + Foto -> (input=text solo con la ruta de la imagen)
* La página debe tener un botón que diga “Generar cv”.
* Una vez presionado el botón, El formulario tiene que ocultarse y se mostrará un CV con toda la información ingresada.
* Una vez renderizado el CV abajo debe aparecer 2 botones:
  + Imprimir” -> imprime el CV generado
  + “Ingresar otro CV” -> volvemos al formulario

**TP12: Mi serie**

Elegir una serie de TV y crear una pequeña página web que tenga:

* Una sección con el encabezado
* Una sección con unas tabs
* Una sección con un acordeón
* 1 footer con un ancla animada para ir arriba de la pagina

**TP13: Blog**

* El blog deberá tener la posibilidad de generar “artículos” por formulario.
* Cada artículo tendrá un Titulo, Fecha, una descripción y una imagen.
* Una vez completado agregarlo al principio del listado
* Al agregar un Articulo se Debera animarse.
* Además, cada Articulo tendrá un botón “X”, el cual lo cerrará.
* Al pasar el mouse por un Articulo, la opacidad de sus hermanos pasará a ser 0.5

**TP14: Primer Ajax**

Crear un archivo .json que contenga una lista de objetos de programas de tv:

* Cada programa de tv tendrá su título, cantidad de capítulos y descripción.
* Traer los objetos a nuestro js utilizando Ajax.
* Renderizar la lista de programas de tv en el html