Certified Tech Developer
The Ultimate Degree

Examen final de Programación Imperativa

¡Llegó el momento de poner a prueba todo lo que estuvimos viendo a lo largo de estas semanas!

Metodología de evaluación

Se evaluarán los siguientes conceptos sobre el código entregado:

- **FORMA**
  - Que el código esté prolijo e implemente buenas prácticas
  - Que las variables, métodos y funciones tengan nombres descriptivos
  - Que utilices nombres en español o en inglés pero no ambos
  - Que utilices camelCase donde corresponda
- **LÓGICA**
  - Que la lógica corresponda con lo que solicitan las consignas
  - Que utilices los métodos más adecuados para cada caso
- **FUNCIONAMIENTO**
  - Que el código funcione correctamente, sin arrojar errores
  - Que el código produzca el resultado esperado a partir de los datos suministrados

Duración, formato y entrega

El examen tendrá una duración de 100 minutos (aprox). La hora de finalización será informada por el docente. ⚠️ **Las entregas realizadas después del tiempo estipulado no serán tenidas en cuenta**.

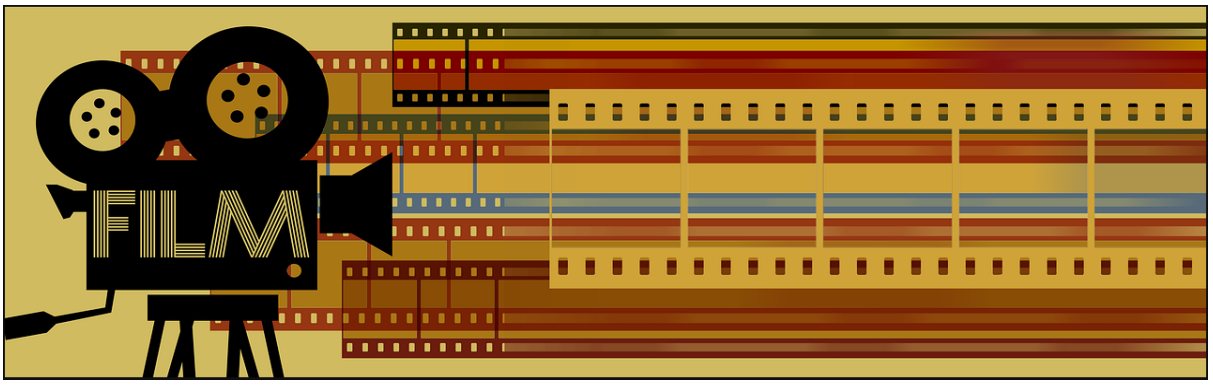
Al terminar el examen, deben entregarlo haciendo uso del formulario que les enviará el docente por chat de zoom.

⚠️**Recuerden verificar la recepción del formulario con el/la docente antes de retirarse del zoom**⚠️

Algunos consejos 🤓👌

- Es normal tener nervios en esta etapa. Recordá que practicamos mucho estas semanas y tratá de mantener la mayor calma que puedas.
- Lee todas las consignas antes de empezar el examen para tener una idea general de lo que se pide.
- Si alguna consigna no queda clara, recordá que podés pedir aclaración por privado al docente.
- Si una consigna te bloquea demasiado, pasá a la siguiente, lo más probable es que luego se te ocurra una solución.

Introducción



Una plataforma de streaming de películas nos contacta para ayudarles a facilitar sus tareas diarias con respecto al manejo de datos.

Consignas

Preparando el entorno...

Descarga esta carpeta con los archivos necesarios, encontrarás un .json con los datos de las películas, un módulo jsonHelper.js con los métodos para leer y escribir en json y una plantilla donde desarrollarás las consignas, con esto deberás:

A. Crear un objeto literal que represente la aplicación.
   El objeto será la representación de nuestro sistema de gestión de películas, podemos llamarlo **gestionDePeliculas** y contendrá todas las propiedades y métodos necesarios.
B. Agregar una propiedad llamada ***peliculas*** en la que asignarás las películas obtenidas a partir del método *leer* del objeto requerido como módulo

(jsonHelper.js), el cual debes requerir como cualquier módulo, al comienzo del archivo, como hemos visto en las prácticas previas.

C. Agregar un método **listarPelículas** que reciba como parámetro un array de películas y los imprima por consola.

   Este método deberá imprimir por consola un mensaje con el siguiente formato:

   {titulo}, de {director}. Duración: {duración} minutos, {genero}, ({calificación} / 10 en IMDB)

   Ejemplos:

   The Mist, de Frank Darabont. Duración: 126 minutos, Horror, (7.1 / 10 en IMDB)

   Tom and Huck, de Peter Hewitt. Duración: 97 minutos, Comedia, (5.5 / 10 en IMDB)

   ⚠️ **Importante:** Este método listarPeliculas deberá ser invocado en cada ejecución de todos aquellos métodos que retornan un array de películas. Digamos que reemplazan al console.log() para hacer más agradable el muestreo de datos.

   Resultado esperado al ejecutar el método: un mensaje por consola por cada película con el formato indicado.

D. Agregar un método **buscarPorId** que permita buscar una película en función de su identificador.
   - Este método recibirá por parámetro un valor de tipo **Number** que representa el **id** a buscar.

- En caso de encontrar una película con el id buscado, devolverá el objeto literal que la representa.
- En caso contrario devolverá *undefined*

E. Agregar un método **peliculasPorGenero**, que retorne todas las películas de determinado género, es decir, aquellas que tengan la propiedad **genero** igual al parámetro recibido.

- Este método recibirá como parámetro un valor de tipo String con el género que se desea.
- Este método debe recorrer el listado de películas completo para filtrar.
- Este método devolverá un array con las películas que sean del género ingresado.

F. Agregar un método **filtrarPorCalificacion** que permita filtrar las películas que tengan una calificación que esté entre el mínimo y máximo enviado.

- Este método recibirá por parámetro dos valores de tipo **Number** que representan la calificación mínima y máxima a buscar.
- Este método devolverá un array con todas las películas que tengan calificación mayor o igual al primer parámetro y menor o igual al segundo parámetro.
- En caso de no encontrar ninguna película, devolverá un array vacío.

G. Agregar un método **ordenarPorDuracion** que permita ordenar las películas recibidas de mayor a menor según su duración.

- El método no recibirá ningún parámetro.
- Este método devolverá un array con todas las películas ordenadas por su duración de mayor a menor.

H. Agregar un método **duracionPromedio** que permita calcular la duración promedio en minutos de las películas.
   - Este método no recibirá ningún parámetro.
   - Este método devolverá un string con el siguiente formato:

     El promedio de duración de las películas es de: {promedio} minutos

I. Agregar un método **modificarGeneroPorId** que permita cambiar el género de una película en función de su id y guardar los cambios en el archivo JSON.

   ⚠️ ***Importante:*** *el método de escritura modifica el archivo JSON original, te recomendamos tener una copia a mano por si algo sale mal y necesitás restaurar el archivo.*

   - Este método recibirá por parámetro un valor de tipo **Number** que representa el **id** a buscar
   - Además recibirá por parámetro un valor de tipo **String** que representa el nuevo **género** que se le asigne a dicha película.
   - Este método utilizará el método **buscarPorId**
   - En caso de encontrar una película con el id buscado:
     i. Cambiará el valor de la propiedad **genero** con aquel recibido por parámetro.

ii. Escribirá los cambios en el archivo JSON que contiene las películas.

- En caso contrario devolverá *undefined*
- Para verificar que el guardado fue correcto, en la sección de ejecuciones de este punto, debes volver a invocar el método que busca por **id** y que se muestre en consola que dicha película fue modificada correctamente y guardada correctamente en el json actualizándolo.

⚠️ **Importante:** el método que te damos para escribir el archivo JSON reemplaza todo el archivo, así que le vas a tener que pasar el array completo de películas y no solo el elemento modificado 😉.