****

# 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



Un restaurante nos contacta para ayudarles a facilitar sus tareas diarias con respecto al manejo de datos.

### **Preparando el entorno...**

Descarga [esta carpeta](https://drive.google.com/drive/folders/1_BBE3U2pFjrK1L-i2SA43qYn118VoD05?usp=sharing) con los archivos necesarios, encontrarás un .json con los datos de las recetas, un módulo jsonHelper.js con los métodos para leer y escribir en json y una plantilla donde desarrollarás las consignas, con esto deberás:

1. Crear un objeto literal que represente la aplicación.   
   El objeto será la representación de nuestro sistema de gestión de recetas, podemos llamarlo **restaurante** y contendrá todas las propiedades y métodos necesarios.
2. Agregar una propiedad llamada ***recetas*** en la que asignarás las recetas 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.
3. Agregar un método **buscarPorNombre** que permita buscar una receta en función de su nombre.
   * Este método recibirá por parámetro un valor de tipo **String** que represente el **nombre** a buscar.
   * En caso de encontrar una receta, devolverá el objeto literal que la representa.
   * En caso contrario devolverá *undefined*
4. Agregar un método **filtrarPorTiempo** que permita filtrar las recetas que tengan un tiempo de espera menor al tiempo enviado como argumento.
   * Este método recibirá por parámetro un valor de tipo **Number** que representa el tiempo de espera máximo.
   * Este método devolverá un array con todas las recetas que tengan un tiempo menor o igual al parámetro enviado
   * En caso de no encontrar ninguna receta que cumpla con esta condición, devolverá un array vacío.
5. Agregar un método **ordenarPorDificultad** que permita ordenar todas las recetas de mayor a menor según su dificultad.
   * Este método no recibirá ningún parámetro
   * Este método retornará un array con todas las recetas ordenadas por su dificultad.
6. Agregar un método **duracionPromedio** que permita calcular la duración promedio en minutos de todas las recetas.
   * Este método no recibirá ningún parámetro.
   * Este método retornará un string con el siguiente formato:

*La duración promedio de todas las recetas es de {tiempo} minutos*

1. Agregar un método **IncrementarPrecio** que permita incrementar él precio de una receta en función de su nombre 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 **String** que represente el **nombre** a buscar
   * Este método utilizará el método **buscarPorNombre**
   * En caso de encontrar una receta con el **nombre** buscado:
     1. Cambiará el valor de la propiedad **precio,** incrementando su valor un 3%.
     2. Escribirá los cambios en el archivo JSON que contiene las recetas.
     3. retornar él objeto modificado
   * En caso contrario devolverá *undefined*

**⚠️ 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 recetas y no solo el elemento modificado 😉.