TA 1

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **repeatString (texto: string, repeticiones:int), que utilizando un loop, imprima el páramentro texto tantas veces como el parámetro repeticiones indique.**

La impresión puede hacerse usando console.log(texto)

TA 2

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **reverseString**(texto: string), que invierta el párametro texto y lo imprima usando console.log().

Pueden usarse diferentes métodos provistos por JavaScript, la idea no es hacer el reverso a mano

TA 3

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **removeFromArray**(arreglo: Array<any>, item: <any>) que remueva el parámetro item del párametro arreglo, e imprima arreglo usado console.log();

Pueden usarse diferentes métodos provistos por JavaScript.

TA 4

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **sumAll**(a: int, b: int) que calcule la suma de todos los números desde el parámetro a al parámetro b (incluídos) y lo imprima usando console.log()

TA 5

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **leapYears**(año: int), que imprima en consola true si el parámetro año es un año bisiesto y false en caso contrario.

Un año bisiesto es aquel que es divisible entre 4, o si es divisible entre 100 y también es divisible entre 400.

TA 6

Dado un archivo HTML, crear un archivo .js que implemente 2 funciones llamadas **convertToCelsius**(temp: int) y **convertToFahrenheit(**temp: int), que haga la conversión de un valor de temperatura de Celsius a Fahrenheit y viceversa, e imprima los resultados en consola.

Nos interese que el resultado sea legible para una persona, por lo que el resultado debe estar redondeado a un valor decimal (ej.: convertToCelsius(100) debe retornar 37.8 y no 37.77777777777778)

TA 7

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **getTheTitles**(books: Array), que devuelva e imprima en consola un nuevo arreglo que contenga sólamente los títulos de los libros guardados en el parámetro books.

El parámetro books debe tener este formato (puede y debe agregar más libros):

```
const books = [
 {
   title: 'Book',
   author: 'Name'
 },
 {
   title: 'Book2',
   author: 'Name2'
 }
]
```

Se pueden usar métodos provistos por JavaScript para realizar este ejercicio.

TA 8

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **findTheOldest**(people: Array), que devuelva e imprima en consola aquella persona dentro del parámetro people que sea el que tenga más edad.

El parámetro people debe tener este formato (puede y debe agregar más personas):

```
const people = [
    {
      name: "Carly",
      yearOfBirth: 1942,
      yearOfDeath: 1970,
    },
```

```
  {
    name: "Ray",
    yearOfBirth: 1962,
    yearOfDeath: 2011,
  },
  {
    name: "Jane",
    yearOfBirth: 1912,
    yearOfDeath: 1941,
  },
]
```

Se pueden usar métodos provistos por JavaScript para realizar este ejercicio.

TA 9

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **getOdds**(nums: Array<int>) que filtre e imprima en consola, en un nuevo arreglo, aquellos números dentro del parámetro nums que sean impares.

No se debe iterar manualmente sobre el parámetro nums, utilice métodos provistos por JavaScript para realizar este ejercicio.

TA 10

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **getSum**(nums: Array<int>) que calcule la suma de todos los numéros dentro del parámetro nums y la imprima en consola.

No se debe iterar manualmente sobre el parámetro nums, utilice métodos provistos por JavaScript para realizar este ejercicio.

TA 11

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **duplicates**(nums: Array<int>) que calcule la cantidad de elementos repetidos dentro del parámetro nums y la imprima en consola.

Ej.: [1, 2, 2, 3, 4, 4, 4, 5] debería devolver 2 (ya que el 2 y el 4 están duplicados)

TA 12

Dado un archivo HTML, crear un archivo .js que implemente una función llamada **generatePassword**(length: int) la cual debe crear una contraseña de largo igual al parámetro length, y que debe cumplir lo siguiente:

- La contraseña debe incluir letras mayúsculas, minúsculas, números y símbolos especiales
- Debe tener un largo mínimo de 8 caracteres
- Llamar, por ejemplo, **generatePassword**(8) dos veces, debería producir resultados diferentes