PRACTICA EN EQUIPO: TYPESCRIPT

INSTRUCCIONES:

- Genere un archivo .zip que contenga el código fuente de su proyecto, y coloque el archivo en el espacio correspondiente de Google Classroom. El archivo debe llamarse MATRICULA1_MATRICULA2_TYPESCRIPT.zip. Solo se debe colocar un archivo por equipo.

1. Genere una carpeta exclusiva para éste proyecto. Posteriormente genere el archivo "tsconfig.json" con el comando correspondiente.

2. (5%) Modifique el archivo "tsconfig.json" de tal forma que soporte ECMA Script a partir de la versión 2019, y que el código compile de forma automática.

3. (5%) Genere una clase llamada "Calculadora" que contenga los métodos necesarios para sumar, restar, multiplicar, y dividir 2 números. Cada método debe recibir un tercer parámetro opcional, con el cual se deberá utilizar para multiplicarlo por el resultado obtenido de la operación. Cada método debe utilizar "Template Literals" para mostrar el resultado en la consola del navegador, indicando el texto de la siguiente forma " La 'operación realizada' de 'numero 1' y 'numero 2' es: 'resultado' ", o " La 'operación realizada' de 'numero 1' y 'numero 2', y multiplicado por 'numero 3' es: 'resultado' ". Ejemplos de la invocación:

```
calculadora.suma(1,2);          La suma de 1 y 2 es: 3
calculadora.suma(1,2,3);        La suma de 1 y 2, y multiplicado por 3 es: 9
calculadora.resta(1,2);         La resta de 1 y 2 es: -1
calculadora.resta(1,2,3);       La resta de 1 y 2, y multiplicado por 3 es: -3
```

4. (40%) Genere una clase llamada "CalculadoraArrow" que contenga la misma funcionalidad que la clase del punto 3, a excepción de que los métodos deben ser Funciones de Flecha:

```
calculadoraArrow.suma(1,2);     La suma de 1 y 2 es: 3
calculadoraArrow.suma(1,2,3);   La suma de 1 y 2, y multiplicado por 3 es: 9
calculadoraArrow.resta(1,2);    La resta de 1 y 2 es: -1
calculadoraArrow.resta(1,2,3);  La resta de 1 y 2, y multiplicado por 3 es: -3
```

5. (50%) Genere una clase llamada "CalculadoraPromise", que contenga la misma funcionalidad que la clase del punto 3, a excepción de que los métodos deben ser asíncronos, utilizando Promises. Utilice la función "setTimeout()" para simular tardanza entre los métodos, de tal forma que cada método tenga un tiempo de ejecución distinto. Un ejemplo de la invocación sería:

```
let calculadoraPromise:CalculadoraPromise = new CalculadoraPromise();
calculadoraPromise.suma(1,2,3);
calculadoraPromise.suma(1,2);
calculadoraPromise.resta(1,2,3);
calculadoraPromise.resta(1,2);
```

```
La suma de 1 y 2, y multiplicado por 3 es: 9
Suma exitosa...
La suma de 1 y 2 es: 3
Suma exitosa...
La resta de 1 y 2, y multiplicado por 3 es: -3
Resta exitosa...
La resta de 1 y 2 es: -1
Resta exitosa...
```