Guia de Ejercicios

Clase I

Multiple Choice

1. ¿Qué implica que un lenguaje sea compilado?

    A. Requieren que un compilador lo “lea”, para revisar que no se hayan cometido errores antes de su ejecución

    B. Eliminan todos los errores de un programa

    C. Un compilador facilita la ejecución rápida del código

2. ¿Cómo se puede imprimir información en pantalla en TypeScript?

    A. console.log()

    B. print()

    C. System.out.println()

¿Cómo se define una constante en TypeScript?

a) Con la palabra clave const

b) Con la palabra clave var

c) Con la palabra clave let

Respuesta: a

¿Cuál es la sintaxis de un bucle for en TypeScript?

a) for (let i = 0; i < array.length; i++) {}

b) for (let i = 0; i < array.length) {}

c) for (i = 0; i < array.length; i++) {}

Respuesta: a

¿Qué es un array en TypeScript?

a) Una variable que almacena un valor único

b) Una estructura de datos que almacena múltiples valores

c) Una constante que no puede ser reasignada

Respuesta: b

¿Cómo se define un array en TypeScript?

a) let myArray = []

b) let myArray = {}

c) let myArray = ()

Respuesta: a

¿Cómo se accede a un elemento de un array en TypeScript?

a) myArray[i]

b) myArray.i

c) myArray.get(i)

Respuesta: a

¿Qué es una función en TypeScript?

a) Una estructura de control que se utiliza para repetir código

b) Una estructura de control que se utiliza para tomar decisiones

c) Un bloque de código que se utiliza para realizar una tarea específica

Respuesta: c

Completar con la palabra correcta

1. let numero: ____ = '1'
2. let numero: ____ = 13
3. let valorDeVerdad: ____ = 'true' // string

4. let valorDeVerdad: ____ = true // boolean

Verdadero o Falso

1. 1 == '01'
2. 1 === '01'
3. NaN === NaN
4. !False
5. !!False
6. !''
7. !!'Hola Mundo!'
8. true == 'true'
9. true == !!'true'

Codigo:

1. Imprimir los números entre 1 y 1000 (1, 2, 3, 4, ...)
2. Imprimir los valores impares que están entre 1 y 1000 (1, 3, 5, 7, 9, ...)
3. Un matemático italiano del siglo XIII, conocido como Fibonacci, describió una sucesión de números. Esta sucesión tiene numerosas aplicaciones en la computación, matemáticas y teoría de juegos, además de aparecer en configuraciones biológicas. La misma se dio a conocer como la solución a un problema de cria de conejos:

Número de mes	Explicación de la genealogía	Parejas de conejos
Comienzo del mes 1	Nace una pareja de conejos (pareja A).	1 pareja en total.
Fin del mes 1	La pareja A tiene un mes de edad. Se cruza la pareja A.	1+0=1 pareja en total.
Fin del mes 2	La pareja A da a luz a la pareja B. Se vuelve a cruzar la pareja A.	1+1=2 parejas en total.
Fin del mes 3	La pareja A da a luz a la pareja C. La pareja B cumple 1 mes. Se cruzan las parejas A y B.	2+1=3 parejas en total.
Fin del mes 4	Las parejas A y B dan a luz a D y E. La pareja C cumple 1 mes. Se cruzan las parejas A, B y C.	3+2=5 parejas en total.
Fin del mes 5	A, B y C dan a luz a F, G y H. D y E cumplen un mes. Se cruzan A, B, C, D y E.	5+3=8 parejas en total.
Fin del mes 6	A, B, C, D y E dan a luz a I, J, K, L y M. F, G y H cumplen un mes. Se cruzan A, B, C, D, E, F, G y H.	8+5=13 parejas en total.
...	...	...
...	...	...

Los números de Fibonacci quedan definidos por las siguientes condiciones iniciales:

$f_0 = 0$

$f_1 = 1$

$f_{n+2} = f_{n+1} + f_n$

Asi:

$f_2 = f_1 + f_0 = 1 + 0 = 1$

$f_3 = f_2 + f_1 = 1 + 1 = 2$

$f_4 = 2 + 1 = 3$

etc

Imprimir los 100 primeros números de la sucesión de Fibonacci (1 1 2 3 5 8 13 21 34 55...).

Recomendación: guardar los últimos 2 valores de f (f_n y f_{n+1}) en variables, ya que de esa forma crearemos el nuevo valor de la función.

Teniendo el arreglo:

let jugadores = ['Pedro', 'Juan', 'Martín', 'Pedro', 'Ignacio', 'Matías', 'Roberto', 'Julián', 'Esteban', 'Pedro', 'Alejandro', 'Franco', 'Francisco', 'Daniel', 'Pedro', 'Manuel', 'Pedro', 'Marco', 'Javier', 'Miguel'];,

4. Imprimir todos los nombres dentro del array, sin importar si se repiten
5. Imprimir las posiciones dentro del arreglo de jugadores donde se encuentra cada Pedro.
6. Realizar una funcion *mayorQueCeroMenorQueDiez* que reciba un numero como argumento y devuelva *true* si un número es mayor que cero y menor que diez, y *false* en caso contrario.

   Indicar, además, el tipo del argumento que recibe y el tipo del valor de retorno:

   function mayorQueCeroMenorQueDiez(numero: ____): _____

Teniendo un arreglo de numeros:

let numeros: number[] = [15,16,3,4,8,10,12,27];

7. Imprimir la suma de dichos numeros
8. Imprimir el producto de dichos numeros
9. Hacer una funcion *dividir* que haga la division entre 2 numeros, si es que el segundo es distinto de cero. En el caso de que sea igual a 0, retornar el valor 0.
10. Hacer una funcion *entre* que reciba 3 numeros, y retorne true si el ultimo esta entre los otros 2. Sino, retorna false