**4-EJERCICIO-TIPO DE DATO**

En este desafío encontrarás una función llamada solution que recibe un parámetro desde la función solution.

Recuerda que el parámetro valor será distinto por cada distinta forma en que ejecutemos la función solution.

Por ejemplo:

Dados los siguientes llamados a la función solution:

solution(1)

solution("Dieguillo")

solution(true)

Debes obtener los siguientes resultados:

"number"

"string"

"boolean"

**7-EJERCICIO-CALCULA LA PROPINA**

En este desafío tendrás que calcular la propina que deben dejar los clientes de un restaurante en función de su consumo.

La función calculateTip recibirá dos parámetros, billAmount que representa el costo total de lo que se haya consumido y tipPercentage que representa el porcentaje de propina a dejar. Ambos valores serán de tipo Number y siempre serán positivos, incluyendo el 0. La función deberá devolver el valor de la propina como un número.

Tendrás inputs y outputs como los siguientes 👇

Ejemplo 1:

Input: calculateTip(100, 10);

Output: 10;

Ejemplo 2:

Input: calculateTip(1524.33, 25);

Output: 3

**11-EJERCICIO-AÑO BISIESTO**

En este desafío, debes crear la lógica de la función isLeapYear, que determina si un año es bisiesto o no. Un año es bisiesto si cumple con las siguientes condiciones:

* Es divisible por 4, pero no por 100.
* Si es divisible por 100 debe serlo por 400 también.

La función isLeapYear recibe un único parámetro: el año a evaluar. Debe devolver true si el año es bisiesto o false en caso contrario.

Toma en cuenta que la función debe ser capaz de manejar valores no enteros o negativos.

Ejemplo 1:

*Input:* 2000;

*Output:* true;

Ejemplo 2:

Input: -2024;

Output: false;

Ejemplo 3:

*Input:* 1984.25;

*Output:* false;

**13-EJERCICIO- Obtén información de mascotas según su tipo**

En este desafío recibirás una serie de tipos de mascotas junto con su edad.

Dependiendo de estos 2 factores tendrás que construir la función llamada getPetExerciseInfo la cual retornará una cadena de texto con la información necesaria acerca de cuanto ejercicio necesita hacer cada tipo de mascota.

La función recibirá las siguientes mascotas:

* perro
* gato
* ave

En caso de pasar una mascota la cual no sea de la lista deberá retornar "Tipo de mascota inválida"

Para cada tipo de mascota, la función retornará diferente información basada en la edad.

* Perros
  + Si la edad es menor al año, deberá retornar "Los cachorros necesitan pequeñas y frecuentes sesiones de juego"
  + Si la edad es entre 1 y 7 años, deberá retornar "Los perros a esta edad necesitan caminatas diarias y sesiones de juego"
  + Si la edad es mayor a 7 años, deberá retornar "Los perros viejos necesitan caminatas más cortas"
* gatos
  + Si la edad es menor al año, deberá retornar "Los gatitos necesitan frecuentes sesiones de juego"
  + Si la edad es entre 1 y 7 años, deberá retornar "Los gatos a esta edad necesitan jugar diariamente"
  + Si la edad es mayor a 7 años, deberá retornar "Los gatos viejos necesitan sesiones de juego más cortas"
* aves
  + Si la edad es menor al año, deberá retornar "Las aves jóvenes necesitan mucho espacio para volar"
  + Si la edad es entre 1 y 7 años, deberá retornar "Las aves necesitan jugar diariamente y un lugar para volar"
  + Si la edad es mayor a 7 años, deberá retornar "Las aves mayores necesitan descansar más, pero siguen ocupando un lugar para volar"

Tendrás inputs y outputs como los siguientes 👇

Ejemplo 1:

Input: getPetExerciseInfo("perro", 3)

Output: "Los perros a esta edad necesitan caminatas diarias y sesiones de juego"

Ejemplo 2:

Input: getPetExerciseInfo("gato", 8)

Output: "Los gatos viejos necesitan sesiones de juego más cortas"

Ejemplo 3:

Input: getPetExerciseInfo("Mamut", 25)

Output: "Tipo de mascota invalida"

**15-EJERCICIO-DIBUJA UN TRIANGULO**

En este desafío, debes dibujar un triángulo isósceles usando bucles.

Recibirás dos parámetros: size y character, que definen el tamaño y el carácter con el que se debe construir el triángulo, respectivamente. Además, el triángulo debe estar alineado a la derecha, lo que significa que la columna más derecha del triángulo debe estar en el borde derecho de la consola.

Recuerda que para hacer el salto de línea debes usar "\n", no olvides removerla de la última parte.

Tendrás inputs y outputs como los siguientes 👇

Ejemplo 1:

Input: printTriangle(5, "\*")

Output:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

Ejemplo 2:

Input: printTriangle(6, "$")

Output:

$

$$

$$$

$$$$

$$$$$

$$$$$$

**17-EJERCICIO-ENCUENTRA MICHI FAMOSO**

En este desafío, debes encontrar al gatito más famoso con base en su número de seguidores.

Recibirás un array de objetos que incluirán las siguientes propiedades:

* name: nombre del gatito.
* followers: un array de números, donde cada uno representa los seguidores de cada red social.

Tu tarea es devolver un array con los nombres de los gatos que tienen solo el mayor número de seguidores. Si hay dos o más gatos con el mismo número máximo de seguidores, deberás incluirlos en el array de resultado, manteniendo el orden en el que aparecen en el array de entrada.

Tendrás inputs y outputs como los siguientes 👇

Ejemplo 1:

Input: findFamousCats([

{

name: "Luna",

followers: [500, 200, 300]

},

{

name: "Michi",

followers: [100, 300]

},

])

Output: ["Luna"]

Ejemplo 2:

Input: findFamousCats([

{

name: "Mimi",

followers: [320, 120, 70]

},

{

name: "Milo",

followers: [400, 300, 100, 200]

},

{

name: "Gizmo",

followers: [250, 750]

}

])

Output: ["Milo", "Gizmo"]