**Reto #0: EL FAMOSO "FIZZ BUZZ”**

/\*

\* Escribe un programa que muestre por consola (con un print) los

\* números de 1 a 100 (ambos incluidos y con un salto de línea entre

\* cada impresión), sustituyendo los siguientes:

\* - Múltiplos de 3 por la palabra "fizz".

\* - Múltiplos de 5 por la palabra "buzz".

\* - Múltiplos de 3 y de 5 a la vez por la palabra "fizzbuzz".

\*/

|  |
| --- |
|  |

#### Reto #1: ¿ES UN ANAGRAMA?

/\*

\* Escribe una función que reciba dos palabras (String) y retorne

\* verdadero o falso (Bool) según sean o no anagramas.

\* - Un Anagrama consiste en formar una palabra reordenando TODAS

\* las letras de otra palabra inicial.

\* - NO hace falta comprobar que ambas palabras existan.

\* - Dos palabras exactamente iguales no son anagrama.

\*/

|  |
| --- |
|  |

#### Reto #2: LA SUCESIÓN DE FIBONACCI

/\*

\* Escribe un programa que imprima los 50 primeros números de la sucesión

\* de Fibonacci empezando en 0.

\* - La serie Fibonacci se compone por una sucesión de números en

\* la que el siguiente siempre es la suma de los dos anteriores.

\* 0, 1, 1, 2, 3, 5, 8, 13...

\*/

|  |
| --- |
|  |

#### Reto #3: ¿ES UN NÚMERO PRIMO?

/\*

\* Escribe un programa que se encargue de comprobar si un número es o no primo.

\* Hecho esto, imprime los números primos entre 1 y 100.

\*/

|  |
| --- |
|  |

#### Reto #4: ÁREA DE UN POLÍGONO

/\*

\* Crea una única función (importante que sólo sea una) que sea capaz

\* de calcular y retornar el área de un polígono.

\* - La función recibirá por parámetro sólo UN polígono a la vez.

\* - Los polígonos soportados serán Triángulo, Cuadrado y Rectángulo.

\* - Imprime el cálculo del área de un polígono de cada tipo.

\*/

#### Reto #5: ASPECT RATIO DE UNA IMAGEN

/\*

\* Crea un programa que se encargue de calcular el aspect ratio de una

\* imagen a partir de una url.

\* - Url de ejemplo: https://raw.githubusercontent.com/mouredev/

\* mouredev/master/mouredev\_github\_profile.png

\* - Por ratio hacemos referencia por ejemplo a los "16:9" de una

\* imagen de 1920\*1080px.

\*/

#### Reto #6: INVIRTIENDO CADENAS

/\*

\* Crea un programa que invierta el orden de una cadena de texto

\* sin usar funciones propias del lenguaje que lo hagan de forma automática.

\* - Si le pasamos "Hola mundo" nos retornaría "odnum aloH"

\*/

#### Reto #7: CONTANDO PALABRAS

/\*

\* Crea un programa que cuente cuantas veces se repite cada palabra

\* y que muestre el recuento final de todas ellas.

\* - Los signos de puntuación no forman parte de la palabra.

\* - Una palabra es la misma aunque aparezca en mayúsculas y minúsculas.

\* - No se pueden utilizar funciones propias del lenguaje que

\* lo resuelvan automáticamente.

\*/

#### Reto #8: DECIMAL A BINARIO

/\*

\* Crea un programa se encargue de transformar un número

\* decimal a binario sin utilizar funciones propias del lenguaje que lo hagan directamente.

\*/

#### Reto #9: CÓDIGO MORSE

/\*

\* Crea un programa que sea capaz de transformar texto natural a código

\* morse y viceversa.

\* - Debe detectar automáticamente de qué tipo se trata y realizar

\* la conversión.

\* - En morse se soporta raya "—", punto ".", un espacio " " entre letras

\* o símbolos y dos espacios entre palabras " ".

\* - El alfabeto morse soportado será el mostrado en

\* https://es.wikipedia.org/wiki/Código\_morse.

\*/

#### Reto #10: EXPRESIONES EQUILIBRADAS

/\*

\* Crea un programa que comprueba si los paréntesis, llaves y corchetes

\* de una expresión están equilibrados.

\* - Equilibrado significa que estos delimitadores se abren y cieran

\* en orden y de forma correcta.

\* - Paréntesis, llaves y corchetes son igual de prioritarios.

\* No hay uno más importante que otro.

\* - Expresión balanceada: { [ a \* ( c + d ) ] - 5 }

\* - Expresión no balanceada: { a \* ( c + d ) ] - 5 }

\*/