

Ejercicios

0.7 Funciones

Introducción

En este tema hemos visto cómo declarar y utilizar funciones. Las funciones son muy útiles para organizar el código, dividirlo en partes manejables, y darle un sentido semántico.

Nos facilitan aplicar la estrategia “divide y vencerás” separando el problema a resolver en problemas más pequeños y manejables.

Además, darle nombres significativos a las funciones ayuda a hacer el código más legible.

Por ejemplo:

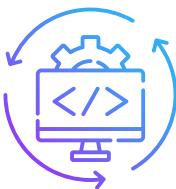
```
1 function imprimirCircunferencia(radio) {  
2   // ... código  
3   console.log(`La circunferencia es ${circunferencia}`);  
4 }  
5  
6 function imprimirAreaCirculo(radio) {  
7   // ... código  
8   console.log(`El área es ${area}`);  
9 }  
10  
11 function describirCirculo(radio) {  
12   imprimirCircunferencia(radio);  
13   imprimirAreaCirculo(radio);  
14 }
```

Entrega

- Cada ejercicio deberá entregarse en un archivo JavaScript (.js) subido a la plataforma del curso.
- La entrega deberá ir acompañada de al menos un diagrama de flujo en formato imagen (se recomienda usar un software de diagramas como Draw.io o LucidChart) que represente el código de la solución de un ejercicio. El ejercicio a escoger se deja a criterio del alumno. Si se entregan más diagramas todos serán corregidos, pero al menos debe entregarse uno. Si no se entrega ningún diagrama la entrega se dará por insatisfactoria.

Evaluación

- Se evaluarán los siguientes puntos:
- Que el código funcione (que no de error).
- Que el código satisfaga el enunciado.
- Que el código no tenga redundancias
- Que el código esté correctamente indentado.
- Que el diagrama sea claro, correcto, y represente el código fielmente.



Ejercicios

0.7 Funciones

Ejercicios

Ejercicio 1

Implementa una función que reciba por parámetro el radio de un círculo y que devuelva la longitud de la circunferencia.

Fórmula: $c = 2\pi r$

Nota: se puede sacar el valor de pi con [Math.PI](#)

Ejemplo:

```
$> node ejercicio1.js
```

```
Introduce el radio: 5
```

```
La circunferencia es 31.41592653589793
```

```
$>
```

Ejercicio 2

Implementa una función que reciba por parámetro el radio de un círculo y que devuelva su área.

Fórmula: $a = \pi r^2$

Ejemplo:

```
$> node ejercicio2.js
```

```
Introduce el radio: 10
```

```
El área del círculo es 314.1592653589793
```

```
$>
```

Ejercicio 3

Implementa una función que reciba por parámetro el radio de un círculo y que pinte por pantalla la circunferencia y el área del círculo.

Ejemplo:

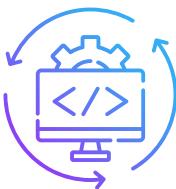
```
$> node ejercicio3.js
```

```
Introduce el radio: 10
```

```
La circunferencia es 62.83185307179586
```

```
El área del círculo es 314.1592653589793
```

```
$>
```



Ejercicios

0.7 Funciones

Ejercicio 4

Implementa una función que reciba por parámetro un array de números y que imprima por pantalla la suma y la media aritmética de sus elementos.

Nota: Se puede dividir un string en un array de elementos con la función [.split\(\)](#)

Ejemplo:

```
$> node ejercicio4.js
```

```
Introduce una lista de números: 10,2,5,33,1,27
```

```
La suma es 78
```

```
El media aritmética es 13
```

```
$>
```

Ejercicio 5

Implementa una función que reciba por parámetro un array de elementos e imprima por pantalla si todos los elementos son únicos en la lista u otra lista con los elementos repetidos en caso contrario.

Ejemplos:

```
$> node ejercicio5.js
```

```
Introduce una lista de elementos: uva,manzana,pera,plátano
```

```
Todos los elementos son únicos
```

```
$>
```

```
-----
```

```
$> node ejercicio5.js
```

```
Introduce una lista de elementos: 1,5,22,5,37,22,2,0
```

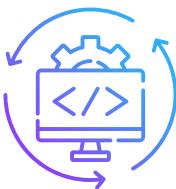
```
Los elementos repetidos son 5, 22
```

```
$>
```

Ejercicio 6

Implementa una función que reciba un número n por parámetro y devuelva una matriz cuadrada n x n con números enteros consecutivos de izquierda a derecha y de arriba a abajo.

Ejemplo:



Ejercicios

0.7 Funciones

```
$> node ejercicio6.js
```

```
Introduce el tamaño de la matriz cuadrada: 3
```

```
[[ 1, 2, 3 ],
 [ 4, 5, 6 ],
 [ 7, 8, 9 ]]
```

```
$>
```

Ejercicio 7

Implementa una función que reciba un texto (string) y que devuelva una matriz bidimensional de ancho 5 y de alto indeterminado que incorpore las palabras (elementos separados por espacios) del texto una por una de izquierda a derecha y de arriba a abajo.

Ejemplos:

```
$> node ejercicio7.js
```

```
Introduce un texto: Mary tenía un corderito, su piel era blanca como la nieve
```

```
[[ "Mary", "tenía", "un", "corderito,", "su" ],
 [ "piel", "era", "blanca", "como", "la"],
 [ "nieve"]]
```

```
$>
```

```
-----
```

```
$> node ejercicio7.js
```

```
Introduce un texto: Soy una tetera
```

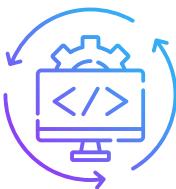
```
[[ "Soy", "una", "tetera" ]]
```

```
$>
```

Ejercicio 8

Implementa una función que reciba un string y devuelva si es o no un palíndromo (independientemente de los espacios).

Ejemplos:



Ejercicios

0.7 Funciones

```
$> node ejercicio8.js
```

Introduce un texto: Mary tenía un corderito

No es palíndromo

```
$>
```

```
$> node ejercicio8.js
```

Introduce un texto: Dabale arroz a la zorra el abad

Sí es palíndromo

```
$>
```

Ejercicio 9

Implementa una función que reciba un número y devuelva una matriz cuadrada en la cual cada elemento es el producto de sus índices.

$$\begin{Bmatrix} 0,0 & 0,1 & 0,2 & 0,3 \\ 1,0 & 1,1 & 1,2 & 1,3 \\ 2,0 & 2,1 & 2,2 & 2,3 \\ 3,0 & 3,1 & 3,2 & 3,3 \end{Bmatrix} \rightarrow \begin{Bmatrix} 0 \times 0 & 0 \times 1 & 0 \times 2 & 0 \times 3 \\ 1 \times 0 & 1 \times 1 & 1 \times 2 & 1 \times 3 \\ 2 \times 0 & 2 \times 1 & 2 \times 2 & 2 \times 3 \\ 3 \times 0 & 3 \times 1 & 3 \times 2 & 3 \times 3 \end{Bmatrix} \rightarrow \begin{Bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 0 & 2 & 4 & 6 \\ 0 & 3 & 6 & 9 \end{Bmatrix}$$

Ejemplo:

```
$> node ejercicio9.js
```

Introduce el tamaño de la matriz cuadrada: 3

```
[[ 0, 0, 0 ],
 [ 0, 1, 2 ],
 [ 0, 2, 4 ]]
```

```
$>
```

Ejercicio 10

Implementa una función que reciba una matriz como parámetro y devuelva la matriz traspuesta.

$$\begin{Bmatrix} a & b & c \\ d & e & f \end{Bmatrix}^T \rightarrow \begin{Bmatrix} a & d \\ b & e \\ c & f \end{Bmatrix}$$



Ejercicios

0.7 Funciones

Ejemplos:

```
$> node ejercicio10.js
```

```
Matriz de entrada
[[ 0, 1, 2 ],
 [ 3, 4, 5 ],
 [ 6, 7, 8 ]]
```

```
Matriz traspuesta
[[ 0, 3, 6 ],
 [ 1, 4, 7 ],
 [ 2, 5, 8 ]]
```

```
$>
```