



**Educación**  
Secretaría de Educación Pública



TECNOLÓGICO  
NACIONAL DE MÉXICO



# Tecnológico Nacional de México

## Campus Felipe Carrillo Puerto

### Ingeniería en Sistemas Computacionales



#### Asignatura

Programación del lado del Servidor

#### Tema

1

#### EVIDENCIA DE APRENDIZAJE

Tarea 1 ejercicios js

#### Profesor

Ing. Paloma Góngora

#### Alumno (s):

Honorio Angulo Pool

#### ISC-8A

Felipe Carrillo Puerto a 13 de enero del 2025



Carretera Vigía Chico SN, C.P. 77200, Colonia Centro, Felipe Carrillo Puerto, Q. Roo  
e-mail: [direcciongitscarrillopuerto.edu.mx](mailto:direcciongitscarrillopuerto.edu.mx) | [tecnm.mx](mailto:tecnm.mx) | [carrillopuerto.tecnm.mx](mailto:carrillopuerto.tecnm.mx)





## Ejercicio 1

```
1.2 Run file > JS index.js
1 // ejercicio 1 Comentarios en línea y multilinea
2 // Este es un comentario en línea. Los comentarios en línea se usan para agregar explicaciones o notas breves
3 // directamente junto a una línea de código. Todo lo que esté después de '//' será ignorado por el intérprete de
4 // JavaScript.
5 console.log('Hola Mundo en nodejs');
6
7 /*
8 Este es un comentario multilinea. Los comentarios multilinea se utilizan cuando se necesita escribir explicacion
9 más largas o detalladas que no caben en una sola línea. Pueden abarcar varias líneas de código y se cierran
10 entre '/*' al inicio y '*/'
11 */console.log('Hola Mundo nodejs');*/
12
13
14 //ejercicio 2 Declarar diferentes tipos de variables y mostrar sus valores por consola.
15 // Declaración de una variable con let
16 let nombre = "Honorio Angulo";
```

## Ejercicio 2

```
1.2 Run file > JS index.js > ...
11 /*console.log('Hola Mundo nodejs');*/
12
13
14 //ejercicio 2 Declarar diferentes tipos de variables y mostrar sus valores por consola.
15 // Declaración de una variable con let
16 let nombre = "Honorio Angulo"; // Una variable que puede ser modificada
17 console.log("El nombre es:", nombre);
18
19 // // Declaración de una variable con const
20 const edad = 30; // Una variable cuyo valor no puede ser modificado
21 console.log("La edad es:", edad);
22
23 //ejercicio 3 Crear un array con al menos cinco elementos de diferentes tipos.
24 // Crear un array con elementos de diferentes tipos
25 /*const miArray = [
26     42, // Número
```



JS index.js

1.2 Run file > JS index.js > ...

```
25 const miArray = [
33     |
34 ];
35
36 // Mostrar los elementos del array en la consola
37 console.log("Elemento 1 (Número):", miArray[0]);
38 console.log("Elemento 2 (Cadena):", miArray[1]);
39 console.log("Elemento 3 (Booleano):", miArray[2]);
40 console.log("Elemento 4 (Objeto):", miArray[3]);
41 console.log("Elemento 5 (Array):", miArray[4]);
42
43 // Llamar a la función dentro del array
44 console.log("Elemento 6 (Función):");
45 miArray[5]();
46
47 // ejercicio 4 Escribe una función que tome dos números y aplique una operación
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

Filter (e.g. text, \exclude, \escape)

🔍 ☰ ^ ×

```
C:\Program Files\nodejs\node.exe .\1.2 Run file\index.js
El nombre es: Honorio Angulo
La edad es: 30
```

The image shows a screenshot of the Visual Studio Code (VS Code) editor interface. The main editor window displays a JavaScript file named `index.js`. The code defines a function `aplicarOperacion` that takes two arguments, `num1` and `num2`, and returns their sum. It also includes comments in Spanish and a log statement to display the result. The code is as follows:

```
49 function aplicarOperacion(num1, num2) {  
55   }  
56  
57 // // Ejemplo de uso de la función  
58 const numero1 = 5;  
59 const numero2 = 10;  
60  
61 // // Llamada a la función y almacenamiento del resultado  
62 const resultadoOperacion = aplicarOperacion(numero1, numero2);  
63  
64 // // Mostrar el resultado en la consola  
65 console.log(`El resultado de la operación es: ${resultadoOperacion}`);  
66  
67 //ejercicio 5 Crea una función flecha que reciba un string y lo imprima en mayúsculas  
68 //const imprimirEnMayusculas = (texto) => {  
69   }  
70 }
```

On the right side of the editor, the **DEBUG CONSOLE** is open, showing the output of the code execution:

```
C:\Program Files\nodejs\node.exe .\1.2 Run file\index.js  
El resultado de la operación es: 15
```

The interface includes a sidebar on the left with tabs for **PROBLEMS**, **OUTPUT**, **DEBUG CONSOLE** (selected), **TERMINAL**, and **PORTS**. The **DEBUG CONSOLE** tab has a search bar with the placeholder text "Filter (e.g. text, exclude, \escape)".



## Ejercicio 5

```
JS index.js X
1.2 Run file > JS index.js > ...
56
57 // // Ejemplo de uso de la función
58 //const numero1 = 5;
59 //const numero2 = 10;
60
61 // // Llamada a la función y almacenamiento del resultado
62 //const resultadoOperacion = aplicarOperacion(numero1, numero2);
63
64 // // Mostrar el resultado en la consola
65 //console.log(`El resultado de la operación es: ${resultadoOperacion}`);
66
67 //ejercicio 5 Crea una función flecha que reciba un string y lo imprima en mayúsculas
68 const imprimirEnMayusculas = (texto) => {
69   console.log(texto.toUpperCase());
70 };
71
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclude, \escape)

C:\Program Files\nodejs\node.exe .\1.2 Run file\index.js  
HONORIO ANGULO

## Ejercicio 6

```
JS index.js X
1.2 Run file > JS index.js > ...
74
75
76
77 // Ejercicio 6 Implementa un bucle que imprima los números del 1 al 10.
78
79 // Bucle for que imprime los números del 1 al 10
80 for (let i = 1; i <= 10; i++) {
81   console.log(i);
82 }
83
84 // Ejercicio 7 Crea un objeto que represente un objeto del mundo real con sus respectivas propiedades
85 // Definimos un objeto que representa un Coche
86 // const coche = {
87 //   marca: "Nissan",
88 //   modelo: "Frontier",
89 // };
90
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclude, \escape)

C:\Program Files\nodejs\node.exe .\1.2 Run file\index.js

```
1
2
3
4
5
6
7
8
9
10
```





## Ejercicio 7

```
JS index.js X
1.2 Run file > JS index.js > [0] coche
87   const coche = {
96     encender: function() {
103   },
104
105   // Método para apagar el coche
106   apagar: function() {
107     if (this.encendido) {
108       this.encendido = false;
109       console.log("El coche está apagado.");
110     } else {
111       console.log("El coche ya está apagado.");
112     }
113   },
114
115   // Método para obtener la información del coche
116   obtenerInformacion: function() {
117     console.log("Marca: Nissan, Modelo: Frontier, Año: 2023, Color: Gris, Kilometraje: 18000 km");
118   }
119 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
C:\Program Files\nodejs\node.exe .\1.2 Run file\index.js
Marca: Nissan, Modelo: Frontier, Año: 2023, Color: Gris, Kilometraje: 18000 km
El coche está encendido.
El coche está apagado.
```

## Ejercicio 8

```
JS index.js X
1.2 Run file > JS index.js > ...
125
126 // Ejercicio 8 Agrega un método al objeto creado anteriormente e imprima una descripción del mismo
127 // Definimos un objeto que representa un Coche
128 const coche = {
129   marca: "Nissan",
130   modelo: "Frontier",
131   año: 2023,
132   color: "Gris",
133   kilometraje: 18000,
134   encendido: false,
135
136   // Método para encender el coche
137   encender: function() {
138     if (!this.encendido) {
139       this.encendido = true;
140       console.log("El coche está encendido.");
141     }
142   }
143 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
C:\Program Files\nodejs\node.exe .\1.2 Run file\index.js
Marca: Nissan, Modelo: Frontier, Año: 2023, Color: Gris, Kilometraje: 18000 km
Este es un Nissan Frontier del año 2023, de color Gris y con un kilometraje de 18000 km.
El coche está encendido.
El coche está apagado.
```





## Ejercicio 9

```
JS index.js X JS app.js
1.2 Run file > JS index.js > ...

179 // Usamos las funciones del módulo
180 const num1 = 10;
181 const num2 = 5;
182
183 console.log(`Suma: ${mathOps.suma(num1, num2)}`);
184 console.log(`Resta: ${mathOps.resta(num1, num2)}`);
185 console.log(`Multiplicación: ${mathOps.multiplicacion(num1, num2)}`);
186 console.log(`División: ${mathOps.division(num1, num2)}`);
187
188 // Manejo de errores en la división
189 try {
190   console.log(`División por cero: ${mathOps.division(num1, 0)}`);
191 } catch (error) {
192   console.log(error.message);
193 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclude, \escape)

C:\Program Files\nodejs\node.exe .\1.2 Run file\app.js

Suma: 15  
Resta: 5  
Multiplicación: 50  
División: 2

## Ejercicio 10

```
JS index.js X
1.2 Run file > JS index.js > operacionAsincronica

196 //ejercicio 10 Escribe una función que simule una operación asincrónica utilizando setTimeout y maneje el resultado
197 function operacionAsincronica(datos, callback) {
198   console.log("Procesando operación asincrónica...");
199
200   setTimeout(() => {
201     // Simulamos el procesamiento de datos
202     const resultado = `Resultado procesado: ${datos}`;
203
204     // Llamamos al callback con el resultado
205     callback(null, resultado); // null indica que no hubo error
206   }, 2000); // Simulación de 2 segundos de retraso
207 }
208
209 // Uso de la función con un callback
210 operacionAsincronica("Datos de prueba", (error, resultado) => {
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, !exclude, \escape) Launch Program

C:\Program Files\nodejs\node.exe .\1.2 Run file\index.js

Procesando operación asincrónica...

Operación completada: Resultado procesado: Datos de prueba





## Ejercicio 11

```
JS index.js x
1.2 Run file > JS index.js > ...
219 function convertirANumero(cadena) {
229     console.log(`La conversión fue exitosa: ${numero}`);
230 } catch (error) {
231     // Maneja cualquier error que ocurra
232     console.log(`Error: ${error.message}`);
233 }
234 }
235
236 // // Ejemplo de uso
237 convertirANumero("123"); // Conversión exitosa
238 convertirANumero("123.45"); // Conversión exitosa
239 convertirANumero("abc"); // Error
240 convertirANumero("123abc"); // Error
241 convertirANumero(""); // Error (cadena vacía)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter (e.g. text, lexclude, \escape) Launch Program
C:\Program Files\nodejs\node.exe .\1.2 Run file\index.js
La conversión fue exitosa: 123
La conversión fue exitosa: 123.45
2 Error: La cadena no es un número válido.
La conversión fue exitosa: 0
```

