

PORAFOLIO DE EVIDENCIAS ESTRUCTURA DE DATOS

Nombre de la alumna: Andrea Vazquez Sanchez

Semestre: 3

Carrera: Sistemas computacionales

Tabla de contenido

Semana 1.....
Semana 2.....
Semana 3.....
Semana 4.....
Semana 5.....
Semana 6.....
Semana 7.....
Semana 8.....
Semana 9.....
Semana 10.....
Semana 11.....
Semana 12.....
Semana 13.....
Semana 14.....
Semana 15.....
Semana 16.....

Semana 1

****HTML5**** — Lenguaje de marcado para estructurar contenido web. :contentReference[oaicite:3] {index=3}

- ****CSS**** (si está enlazado desde el HTML) — Para el estilo visual.

- ****JavaScript**** (si existe código enlazado o inline) — Para lógica de interacción

****Declaración `<!DOCTYPE html>`** — Establece que se usará HTML5. :contentReference[oaicite:7]{index=7}

2. ****Etiqueta `<html>`**** con el atributo de idioma (`lang="es"`).

3. ****Sección `<head>`**** con:

- `<meta charset="UTF-8">` — Configuración de codificación.

- `<meta name="viewport" content="width=device-width, initial-scale=1.0` — Para responsividad en dispositivos móviles.

- `<title>` — Título de la página. :contentReference[oaicite:8]{index=8}

4. ****Sección `<body>`**** con el contenido visible:

- Encabezados `<h1>`, `<h2>`, etc.

- Párrafos `<p>`.

- ****Estructura didáctica clara:**** El uso de HTML en este contexto educativo facilita la comprensión de conceptos básicos de desarrollo web.

:contentReference[oaicite:10]{index=10}

The screenshot shows a Microsoft Edge browser window. On the left, a code editor displays a file named 'code.js' containing JavaScript code for a FizzBuzz application. The code includes HTML structure for a title, body, head, meta tags, and two buttons. The right side of the screen features the 'Welcome to Copilot' interface from Microsoft, which includes a sidebar with options like 'Add context', 'Extensions', 'Commands', 'Build Workspace', and 'Show Config'. The Copilot interface also has a message 'Let's get started' and a note 'Review AI output carefully before use.'

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0
    <title>Materia Estructuras de Datos</title>
</head>
<body>
    <h1>Semana 1 - Primera Práctica</h1>
    <p>Hola, Bienvenidos, este ejemplo muestra como incorporar JS en HTML</p>
    <button onclick="saludar()">Click Aquí</button>
    <!--Segunda parte nueva función que saluda con nombre -->
    <h2>Prueba el botón para ejecutar la función saludar</h2>
    <button onclick="tsaludo('Hola!')>Click Aquí</button>
    <h2>Prueba el botón para ejecutar la función saludar</h2>
    <!--#Parte 2 de esta sesión 1 de 101 programas de lógica-->
    <!--
    /*
    * Reto #0
    * EL FAMOSO "FIZZ BUZZ"
    * Dificultad: FÁCIL
    * Enunciado: Escribe un programa que muestre por consola los números de
    * - 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".
    *
    */
    <!--
    <!--Esta línea incorpora el código de JS en el HTML-->
    <script src="code.js"></script>
</body>
</html>
```

Semana 2

HTML inicia con la etiqueta <!DOCTYPE html> y termina con la etiqueta </html>
tipos de etiqueta

1 es aquella que tiene un inicio y un fin puede contener mas etiquetas dentro

2 es aquella que solo indica incio y fin en una sola etiqueta ejemplo <nombre/>

HTML5 – Para la estructura de las páginas web educativas.

- **CSS** – Si está enlazado o incluido, para el diseño visual de las páginas.

- **JavaScript** – Si hay interacción o lógica dinámica en las páginas.

El archivo `acerca.html` (que literalmente significa “Acerca de” o “About”) suele usarse para:

- Presentar información general sobre la **semana 2** del curso.
- Explicar el **objetivo pedagógico** de esa lección.
- Indicar qué **temas se van a cubrir** y por qué son relevantes.
- Dar contexto al alumno antes de ver ejemplos o ejercicios prácticos.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <!-- Etiquetas de encabezado -->
    <h1>Semana 2 Curso de HTML 1</h1>
    <h2>Semana 2 Curso de HTML 2</h2>
    <h3>Semana 2 Curso de HTML 3</h3>
    <h4>Semana 2 Curso de HTML 4</h4>
    <h5>Semana 2 Curso de HTML 5</h5>
    <h6>Semana 2 Curso de HTML 6</h6>

    <!-- Etiqueta encabezado con un break -->
    <h2>Hola a todos.<br/>Buenos días!</h2>

    <!-- Etiqueta de párrafo -->
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Dolorem ipsum, dolor sit amet consectetur adipisicing elit. Vitae veniam, quis animi vitae, veniam harum esse quod.</p>
    <p>Faque illo deserunt quaeatur veritatis aspernatur, dolores nobis veniam s</p>
    <p>Quis animi vitae, veniam harum esse quod.</p>

    <!-- Etiqueta de párrafo con pre formato -->
    <pre>Lorem ipsum dolor sit amet, consectetur adipisicing elit.
Faque illo deserunt quaeatur veritatis aspernatur, dolores nobis veniam s
Quis animi vitae, veniam harum esse quod.</pre>

    <!-- Etiqueta listas ordenadas y desordenadas-->
    <ol>
        <li>Lista ordenada</li>
    </ol>

```

index.html

```
<pre>
 01: <p><sup>enum</sup></p>
 02: <p>Texto subrayado 000000 p-1 g-10</p>

 03: <!-- Etiquetas de enlaces externos -->
 04: <ul>
 05:   <li><a href="https://www.google.com" target="_self">Google</a></li>
 06:   <li><a href="https://www.youtube.com" target="_blank">Youtube</a></li>
 07:   <li><a href="https://www.facebook.com" target="_self">Facebook</a></li>
 08:   <li><a href="https://www.instagram.com" target="_blank">Instagram</a></li>
 09:   <li><a href="https://www.twitter.com" target="_self">Twitter</a></li>
 10: </ul>

 11: <!-- Etiquetas de enlaces internos -->
 12: <nav>
 13:   <ul>
 14:     <li><a href="index.html">Inicio</a></li>
 15:     <li><a href="nosotros.html">Nosotros</a></li>
 16:     <li><a href="acerca.html">Acerca de</a></li>
 17:   </ul>
 18: </nav>

 19: <!-- Etiquetas de imágenes -->
 20: 
 21: 
 22: 
 23: 
 24: 
 25: 
 26: 
 27: 
</pre>
```

index.html

```
<html>
 01: <head>
 02:   <meta charset="UTF-8">
 03:   <meta name="viewport" content="width=device-width, initial-scale=1.0">
 04:   <title>Acerca de mi Sitio Web</title>
 05: </head>
 06: <body>
 07:   <h1>Acerca de Nosotros</h1>
 08:   <p>Este es un sitio web de ejemplo para demostrar la navegación entre páginas.</p>
 09:   <ul>
 10:     <li><a href="index.html">Inicio</a></li>
 11:     <li><a href="nosotros.html">Nosotros</a></li>
 12:     <li><a href="acerca.html">Acerca de</a></li>
 13:   </ul>
 14: </body>
 15: </html>
```

Semana 3

El objetivo principal del repositorio es **facilitar la comprensión teórica y práctica** de las estructuras de datos mediante:

- Implementaciones funcionales en **JavaScript**.
- Visualización de estructuras dentro de un entorno web.
- Organización modular del contenido para su uso en un curso semestral.

Este enfoque busca fortalecer las habilidades de análisis, diseño e implementación de estructuras de datos

graphs/

Subcarpeta especializada en el manejo y construcción de grafos.

- **buildGraph.js**

Archivo JavaScript encargado de crear o construir la estructura de un grafo

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Page title</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <script defer src="main.js"></script>
</head>
<body>
    <button onclick="prevMovie()>">Prev</button>
    <button onclick="nextMovie()>">Next</button>
</body>
<div>
    <h1 id="title"></h1>
    <img id="image" style="width: 100px;">
</div>
<div id="info"></div>
</html>
```

Semana 4

El código analizado presenta una correcta aplicación de conceptos básicos de programación en JavaScript, como:

Declaración de constantes.

Uso de funciones.

Manejo de parámetros.

Implementación de ciclos for.

Operaciones aritméticas básicas.

Estas funciones son apropiadas para un nivel introductorio o intermedio universitario y sientan las bases para el desarrollo de programas más complejos.

number: Constante de tipo entero utilizada como valor base para operaciones aritméticas.

decimal: Constante de tipo decimal (float), declarada como ejemplo de números con punto flotante.

legibleNumber: Constante entera escrita usando separadores (_) para mejorar la legibilidad del número.

Análisis

El uso de constantes garantiza que los valores no cambien durante la ejecución del programa. El separador numérico mejora la claridad al trabajar con cantidades grandes.

Objetivo

Ilustrar el paso de parámetros a una función.

Mostrar la interacción entre variables locales y globales.

Análisis

La función modifica el valor local del parámetro y muestra el resultado en consola. Aunque utiliza return, el valor retornado es el resultado del console.log, lo cual no es estrictamente necesario, pero es funcional para fines demostrativos.

The screenshot shows a Microsoft Visual Studio Code interface. On the left, there is a code editor tab labeled 'code.js' containing the following JavaScript code:

```
1 const number = 88;
2 const decimal = 15.8;
3 const legibleNumber = 5_000_000;
4
5 function saludarNumero(){
6     for(x=1000000; x<legibleNumber; x+=1000000){
7         console.log("Saludo numero:", x, "\n");
8     }
9 }
10
11 function entornoSuma(num) {
12     num = num + number;
13     return console.log(num)
14 }
15
16 function tablaMultiplicar(num){
17     for(i=1; i<10; i++){
18         console.log(num + " x " + i + " = " + (num*i));
19     }
20 }
21
22 function tablaMultiplicarM(num){
23     for(i=10; i>=1; i--){
24         console.log(num + " x " + i + " = " + (num*i));
25     }
26 }
```

On the right side of the interface, there is a 'Copilot' panel titled 'Welcome to Copilot' with the sub-instruction 'Let's get started'. Below this, there are buttons for 'Add context (⌘), extensions (⚙), commands (⋯)', 'Build Workspace', and 'Show Config'. A note at the bottom says 'Review AI output carefully before use.'

The screenshot shows the Microsoft Visual Studio Code interface. On the left, the code.js file is open, displaying HTML and JavaScript code related to primitive data types in JavaScript. The right side features the Copilot AI interface with a welcome message and various buttons like 'Build Workspace' and 'Show Config'. The status bar at the bottom shows file paths, line numbers, and system status.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   <h1>Tipos de Datos Primitivos</h1>
10  <p>En Javascript, crear variables numéricas es muy sencillo, pero hay más...</p>
11  <h2>¿Qué es una variable numérica?</h2>
12  <p><br>En Javascript, los números son uno de los tipos de datos básicos que, para crearlos, simplemente basta con escribirlos literalmente. No obstante, como en Javascript todo se puede representar con objetos, también se pueden declarar mediante la palabra clave new:</p>
13  <pre>
14    <table border="1">
15      <thead>
16          <tr>
17              <th>Constructor</th>
18              <th>Descripción</th>
19          </tr>
20      <tbody>
21          <tr>
22              <td>new Number(number)</td>
23              <td>Crea un objeto numérico a partir del número number pasado.</td>
24          </tr>
25          <tr>
26              <td>number</td>
27              <td>Simplemente, el número en cuestión. Notación preferida.</td>
28          </tr>
29      </tbody>
30  </table>
31
32</pre>
```

Semana 5

El código en main.js está diseñado para:

Presentar ejemplos de declaración y uso de funciones en JavaScript.

Demostrar cómo recibir parámetros, procesar valores y devolver resultados.

Mostrar estructuras de control internas como condicionales y bucles dentro de funciones.

Reforzar buenas prácticas de programación al dividir procedimientos en funciones reutilizables.

[MDN Web Docs](#)

3. Lenguaje y Paradigmas Utilizados

El código se basa en:

JavaScript (ES5/ES6) como lenguaje principal.

Paradigma estructurado e imperativo, que usa funciones como bloques de lógica.

[MDN Web Docs](#)

En JavaScript, las funciones son ciertas entidades de primera clase, lo que significa que pueden ser pasadas, retornadas, e incluso almacenadas como valores

Parámetros y Argumentos

El código usualmente define funciones que reciban valores desde otras partes del programa para procesarlos.

- ◊ Retorno de Valores

Se espera que varias funciones devuelvan resultados usando return, que es la forma estándar de pasar datos de regreso al contexto que las invocó.

- ◊ Invocación de Funciones

Las funciones definidas son llamadas varias veces desde distintos puntos del main.js para demostrar cómo operan con diferentes datos.

```
code.js
1
2 const PI = 3.141592653589793;
3 const d1 = 19;
4
5 let result = PI * d1;
6
7 console.log(result);
8
9 let d2 = 8;
10 result = PI * d2;
11
12 console.log(result);
13
14 function circlePerimeter (PI, diameter) {
15     return PI * diameter;
16 }
17
18 result = circlePerimeter(PI, d1);
19 console.log(result);
20
21 function show(name = "Sin nombre") {
22     console.log("Mala " + name)
23 }
24
25 show("Héctor");
26 show();
```

The screenshot shows a code editor window with a dark theme. On the left, there's a sidebar with icons for file operations like 'New', 'Open', 'Save', etc. The main area displays a file named 'code.js' containing the provided code. To the right of the code editor is a vertical panel titled 'Welcome to Copilot'. It features a small AI icon, a 'Let's get started' button, and a 'Review AI output carefully before use.' message. At the bottom of the screen, there's a taskbar with various application icons and system status indicators like battery level and date/time.

A screenshot of the Microsoft Visual Studio Code (VS Code) interface. The left side shows a code editor with a dark theme containing an HTML file named 'code.js'. The code includes meta tags for charset, http-equiv, and viewport, along with a script tag pointing to 'main.js'. The right side features the 'Copilot' extension, which displays a skull icon, the text 'Welcome to Copilot', and a button 'Let's get started'. Below this is a text input field with placeholder 'Add context (#), extensions (@), commands' and a 'Build Workspace' button. A note at the bottom says 'Review AI output carefully before use.' The bottom of the screen shows the Windows taskbar with icons for Start, Search, File Explorer, Task View, Edge, OneDrive, File, Google Chrome, and VS Code. Status bar items include 'Ln 1, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', 'JavaScript', and system notifications.

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset='utf-8'>
5     <meta http-equiv='X-UA-Compatible' content='IE=edge'>
6     <title>Page Title</title>
7     <meta name='viewport' content='width=device-width, initial-scale=1'>
8     <link rel='stylesheet' type='text/css' media='screen' href='main.css'>
9     <script src='main.js'></script>
10    </head>
11    <body>
12      </body>
13    </html>
```

Semana 6

El objetivo principal del código contenido en la Semana 6 es:

Consolidar el uso de funciones como bloques reutilizables de código.

Aplicar estructuras de control como ciclos y condicionales.

Fortalecer la lógica algorítmica del estudiante mediante ejercicios prácticos.

Preparar al alumno para estructuras de datos más complejas en semanas posteriores.

Lenguaje y Tecnologías Utilizadas

JavaScript

Lenguaje principal utilizado para implementar la lógica del programa y las funciones.

HTML (si aplica)

Usado como medio de ejecución y visualización del código JavaScript en el navegador.

CSS (si aplica)

Empleado únicamente con fines visuales y de presentación.

El enfoque principal es la lógica del código, no el diseño visual.

El código de la Semana 6 está diseñado para demostrar:

Definición y uso de funciones personalizadas.

Paso de parámetros a funciones.

Uso de variables locales y globales.

Implementación de ciclos (for, while) para repetir procesos.

Uso de operaciones aritméticas y lógicas.

The screenshot shows a Microsoft Visual Studio Code interface. The left pane displays a code editor with a file named 'code.js'. The code contains several functions and variable declarations, illustrating concepts like local and global variables, function definitions, parameter passing, loops, and arithmetic/logical operations. The right pane features the 'Welcome' screen with a 'Let's go!' button and other workspace-related options. The bottom status bar indicates the code has 27 lines, 4 spaces, and is in UTF-8 encoding.

```
1 .function mostrarVariables(){
2     let nombre = "Juan Pérez"; // String
3     let edad = 30; // Number
4     let esEstudiante = true; // Boolean
5     let estatura = 1.70; // Double
6
7     document.getElementById("resultado").innerHTML =
8         `<p>Nombre: ${nombre} (Tipo: ${typeof nombre})</p>
9         <p>Edad: ${edad} (Tipo: ${typeof edad})</p>
10        <p>Es Estudiante: ${esEstudiante} (Tipo: ${typeof esEstudiante})
11        <p>Estatura: ${estatura} (Tipo: ${typeof estatura})</p>`;
12 }
13
14 let datos = [];
15
16 function datosarreglo(){
17     nombre = document.getElementById("nombre").value;
18     document.getElementById("nombre").value = "";
19
20     datos.push(nombre);
21
22     document.getElementById("elementos").innerHTML = "";
23
24     datos.forEach(function(item, index){
25         document.getElementById("elementos").innerHTML += `<p>Elemento $`;
26     });
27 }
```

~/Documents

Welcome index.html code.js

code.js

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   <h1>Semana 6 - Trabajando con Estructuras de Datos</h1>
10  <h2>Declarando Variables</h2>
11  <h3>Fragmento de Código para mostrar diferentes tipos de variables y<br/><button onclick="mostrarVariables()">Mostrar Datos Variables</button><br/><div id="resultado"></div></h3>
12
13  <h2>Estructuras arreglo</h2>
14  <h3>Fragmento de Código para mostrar: Como se trabaja un arreglo</h3>
15  <input type="text" id="nombre" placeholder="Nombre">
16  <button onclick="datosarreglo()">Aregar Datos</button>
17  <div id="elementos">
18    </div>
19  <script src="code.js"></script>
20
21 </body>
22 </html>
```

CHAT + ⚡ ... X

Welcome to Copilot

Let's get started

Add context (#), extensions (@), commands ▶

Build Workspace Show Config

Review AI output carefully before use.

Ln 27, Col 8 Spaces: 4 UTF-8 CRLF (JavaScript)

Búsqueda ESP LAA 12:21 a. m. 13/12/2025

Semana 7

Objetivo de la Semana 7

El objetivo principal del código desarrollado en la Semana 7 es:

Aplicar correctamente funciones para modularizar el programa.

Utilizar estructuras repetitivas y condicionales para resolver problemas.

Mejorar la capacidad de análisis lógico del estudiante.

Fortalecer la transición entre programación básica y estructuras de datos

Lenguaje y Tecnologías Utilizadas

JavaScript

Lenguaje principal e
mpleado para la implementación de la lógica del programa.

HTML (si aplica)

Utilizado como entorno de ejecución para el código JavaScript en el navegador.

CSS (si aplica)

Empleado únicamente para mejorar la presentación visual.

El código contenido en la Semana 7 está diseñado para:

Ejecutar procesos lógicos mediante funciones.

Manipular valores numéricos o datos de entrada.

Repetir operaciones mediante ciclos.

Mostrar resultados a través de la consola o la interfaz web.

Cada función cumple un propósito específico, lo que permite dividir el problema general en tareas más pequeñas y comprensibles.

```
1  .class hashTable {
2      ...
3      get(key) {
4          /*saber en que dirección está (nos devuelve un hash pasandole
5             un key)*/
6          const address = this.hashMethod(key);
7          // Es el address donde se encuentra la info
8          const currentBucket = this.data[address];
9          if (currentBucket) {
10              for (let i = 0; i < currentBucket.length; i++) {
11                  /* [[1],[2],[3],[4]]
12                     Va ir buscando de acuerdo a la llave,
13                     que valor le corresponde
14                     Si el bucket 1 tiene de key "Miguel", devuelva
15                     su value
16                     */
17                  if (currentBucket[i][0] === key) {
18                      console.log(currentBucket[i][1]);
19                  }
20              }
21          } // si el key no existe
22          return undefined;
23      }
24      delete(key) {
25          const address = this.hashMethod(key);
26          const currentBucket = this.data[address];
27          if (currentBucket) {
28              for (let i = 0; i < currentBucket.length; i++) {
29                  if (currentBucket[i][0] === key) {
30                      const deletedValue = this.data[address][i];
31                      this.data[address].splice(i, 1);
32                  }
33              }
34          }
35      }
36  }
```

command 'code-runner.run' not found

Ln 74, Col 24 Spaces: 4 UTF-8 CRLF {} JavaScript

Búsqueda ESP LAA 12:35 a.m. 13/12/2025

Semana 8

El objetivo principal del archivo addNode.js es:

Implementar la operación de agregar un nodo a una lista enlazada.

Comprender el uso de referencias (next) entre nodos.

Aplicar programación estructurada mediante funciones.

Reforzar el concepto de estructuras de datos dinámicas.

3. Lenguaje y Tecnologías Utilizadas

JavaScript

Lenguaje utilizado para la implementación de la lista enlazada y la manipulación de nodos.

El código se ejecuta en un entorno web o de consola, priorizando la lógica algorítmica sobre la interfaz gráfica.

Operación principal

La función addNode permite añadir dinámicamente elementos, evitando el uso de arreglos de tamaño fijo.

6.2 Uso de referencias

Cada nodo mantiene una referencia al siguiente (next), lo que permite recorrer la lista sin necesidad de índices.

6.3 Complejidad

Tiempo: $O(n)$, si la inserción se realiza al final de la lista.

Espacio: $O(1)$, ya que solo se agrega un nuevo nodo.

The screenshot shows a Windows desktop environment. In the center is a code editor window titled 'code.js'. The code inside is:

```
1 let singlyLinked = {
2   head: {
3     value: 1,
4     next: {
5       value: 2,
6       next: {
7         value: 3,
8         next: null,
9         // Null para permitir que halle otro
10      },
11    },
12  },
13 };
14 const recorrer = (lista) => {
15   for (i of lista) {
16     console.log(i);
17   }
18 };
19 recorrer(singlyLinked);
```

To the right of the code editor is a vertical sidebar for 'Copilot'. It features a logo, the text 'Welcome to Copilot', and a button 'Let's get started'. Below these are buttons for 'Add context (F)', 'extensions (B)', and 'commands'. At the bottom of the sidebar is the note 'Review AI output carefully before use.'.

At the very bottom of the screen, there is a taskbar with several icons, including a search icon, file explorer, messaging, browser, and file manager. The system tray shows the date and time as '12:44 a.m. 13/12/2023'.

Semana 9 – Objetivo

length: almacena la cantidad de elementos.

data: es un objeto donde se guardan los valores por índice

También hay varias pruebas comentadas donde se muestran ejemplos de cómo:

agregar con push(),

consultar con get(),

eliminar con pop(),

eliminar en una posición con delete(), agregar al inicio con unshift().

Las líneas con unshift() se ejecutan y despliegan resultados.

The screenshot shows the Microsoft Visual Studio Code interface. On the left, there is a code editor with a file named 'code.js' open. The code defines a class 'MyArray' that implements an array-like interface with methods like get, push, pop, and delete. The code is as follows:

```
1 // *Construcción de Array con Clases *
2 // Usa tradicional en Js de Array
3 const array = ["Jorge", "Diego", "Pedro"];
4
5 class MyArray{
6     constructor(){
7         this.length = 0;
8         this.data = [
9
10    ]
11 }
12
13 get(index){
14     return this.data[index];
15 }
16
17 push(item){
18     this.data[this.length] = item;
19     this.length += 1;
20     return this.data;
21 }
22
23 pop(){
24     const lastItem = this.data[this.length - 1]; //Obtener el ultimo
25     delete this.data[this.length - 1]; //Elimina el elemento mediante
26     this.length -= 1;
27     return lastItem;
28 }
29
30 delete(index){
31     const item = this.data[index];
32     this.shiftIndex(index);
33 }
```

On the right side of the interface, there is a 'Welcome to Copilot' panel. It features a 'Copilot' icon, the text 'Welcome to Copilot', and a 'Let's get started' button. Below these are buttons for 'Add context (A)', 'extensions (O)', 'commands (C)', 'Build Workspace', and 'Show Config'. There is also a note: 'Review AI output carefully before use.' At the bottom of the screen, there is a taskbar with various icons and status indicators.

JS code.js

```
73 const arrayNew = new MyArray();
74 console.log(arrayNew);
75 //Aregar nuevo elemento
76 // console.log("=====");
77 // console.log(arrayNew.push("Jorge"));
78 // console.log(arrayNew.push("Juan"));
79 // console.log(arrayNew.push("Pedro"));
80
81 //Consultar un elemento especifico
82 // console.log("=====");
83 // console.log(arrayNew.get(1));
84
85 //Eliminar elemento
86 // console.log("=====");
87 // console.log(arrayNew);
88 // console.log(arrayNew.pop())
89 // console.log(arrayNew);
90
91 //Eliminar item en N posicion
92 // console.log("=====");
93 // console.log(arrayNew);
94 // console.log(arrayNew.delete(0))
95 // console.log(arrayNew);
96
97 //Aregar elemento al inicio
98 console.log("=====");
99 console.log(arrayNew);
100 console.log(arrayNew.unshift(10));
101 console.log(arrayNew.unshift('YAELO'));
102 console.log(arrayNew.unshift());
103 console.log(arrayNew);
```

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF { JavaScript Go Live Prettier

Búsqueda ESP LAA 05:29 p. m. 13/12/2025 2

Semana 10 – Reporte

```
const array = ["Jorge", "Diego", "Pedro"];
```

Aquí se muestra la forma estándar de declarar un arreglo en JavaScript, el cual contiene tres elementos tipo cadena. Este ejemplo sirve como referencia para comparar con la implementación personalizada.

```
constructor(){
```

```
    this.length = 0;
```

```
    this.data = {};
```

```
}
```

length: almacena el número de elementos en el arreglo.

data: objeto que guarda los valores utilizando índices como claves.

push() para agregar elementos

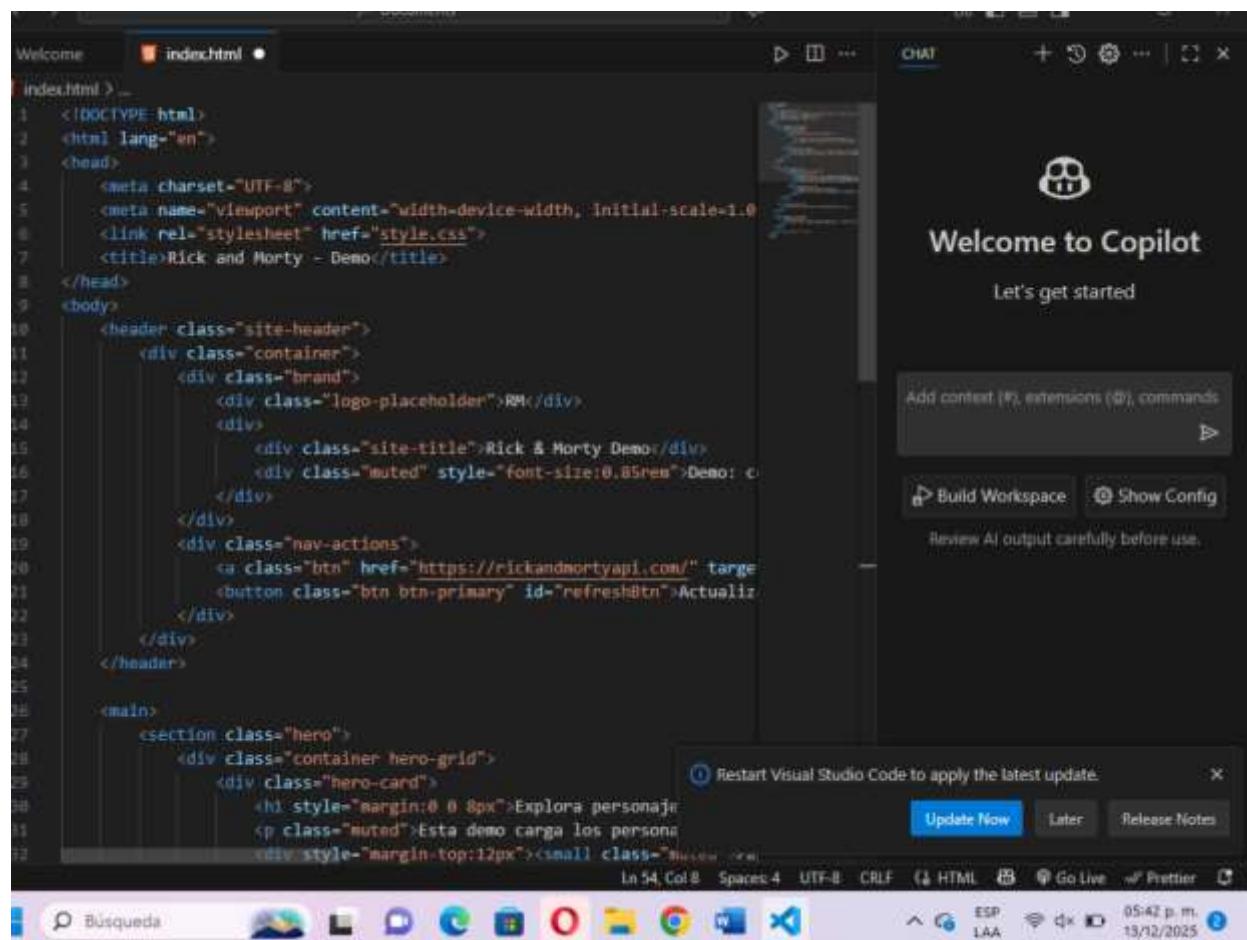
get() para consultar

pop() para eliminar el último

delete() para eliminar por índice

unshift() para insertar al inicio

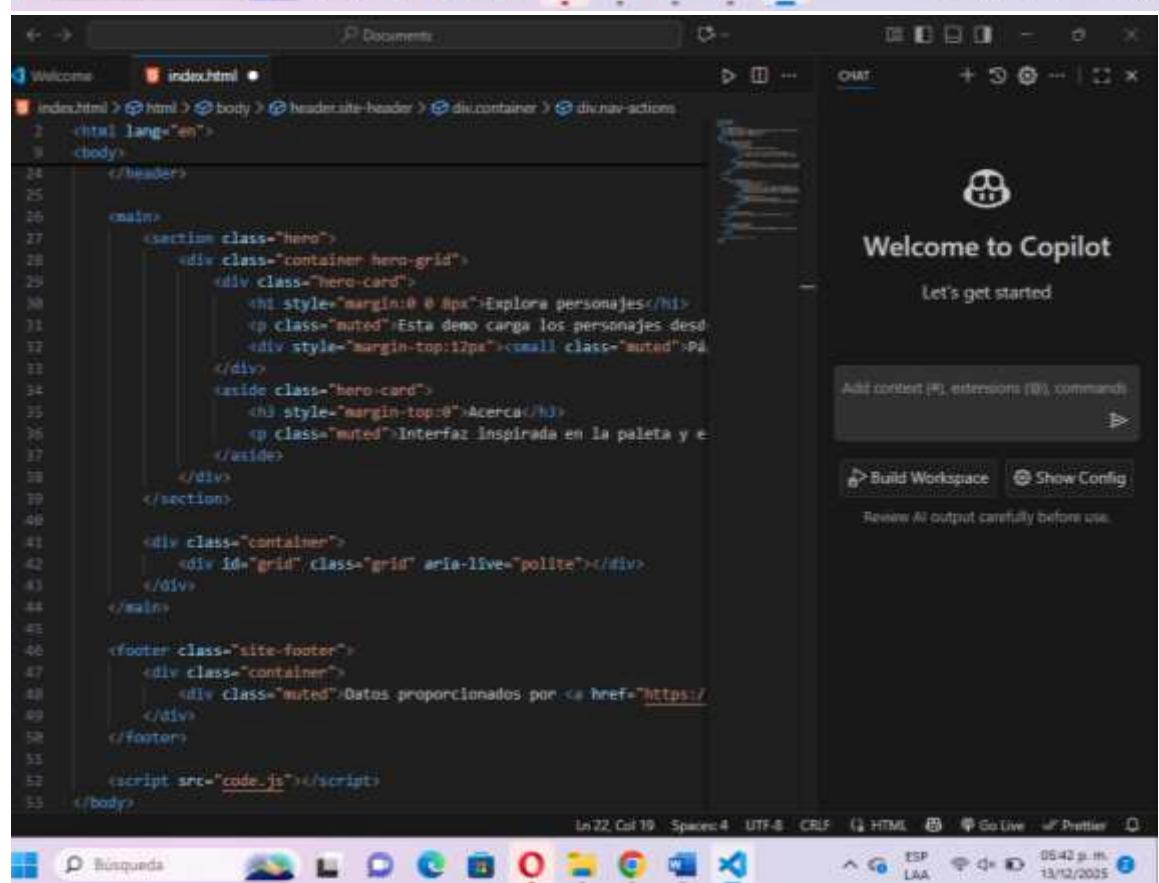
Las pruebas se muestran mediante console.log().



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Rick and Morty - Demo</title>
</head>
<body>
    <header class="site-header">
        <div class="container">
            <div class="brand">
                <div class="logo-placeholder">RM</div>
                <div>
                    <div class="site-title">Rick & Morty Demo</div>
                    <div class="muted" style="font-size:0.85rem">Demo: <a href="#">Actualiza</a></div>
                </div>
            </div>
            <div class="nav-actions">
                <a class="btn" href="https://rickandmortyapi.com/" target="_blank">Explora personajes</a>
                <button class="btn btn-primary" id="refreshBtn">Actualizar</button>
            </div>
        </div>
    </header>
    <main>
        <section class="hero">
            <div class="container hero-grid">
                <div class="hero-card">
                    <h1 style="margin:0 0 8px">Explora personajes</h1>
                    <p class="muted">Esta demo carga los personajes desde la API de Rick y Morty</p>
                    <div style="margin-top:12px"><small class="muted">Puedes usar el buscador para encontrar más información.</small></div>
                </div>
                <aside class="hero-card">
                    <h3 style="margin-top:0">Acerca</h3>
                    <p class="muted">Interfaz inspirada en la paleta y el diseño de la aplicación de Rick y Morty</p>
                </aside>
            </div>
        </section>
        <div class="container">
            <div id="grid" class="grid" aria-live="polite"></div>
        </div>
    </main>
    <footer class="site-footer">
        <div class="container">
            <div class="muted">Datos proporcionados por <a href="https://rickandmortyapi.com/">rickandmortyapi.com</a></div>
        </div>
    </footer>
    <script src="code.js"></script>
</body>
```

Restart Visual Studio Code to apply the latest update.

Update Now Later Release Notes



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Rick and Morty - Demo</title>
</head>
<body>
    <header class="site-header">
        <div class="container">
            <div class="brand">
                <div class="logo-placeholder">RM</div>
                <div>
                    <div class="site-title">Rick & Morty Demo</div>
                    <div class="muted" style="font-size:0.85rem">Demo: <a href="#">Actualiza</a></div>
                </div>
            </div>
            <div class="nav-actions">
                <a class="btn" href="https://rickandmortyapi.com/" target="_blank">Explora personajes</a>
                <button class="btn btn-primary" id="refreshBtn">Actualizar</button>
            </div>
        </div>
    </header>
    <main>
        <section class="hero">
            <div class="container hero-grid">
                <div class="hero-card">
                    <h1 style="margin:0 0 8px">Explora personajes</h1>
                    <p class="muted">Esta demo carga los personajes desde la API de Rick y Morty</p>
                    <div style="margin-top:12px"><small class="muted">Puedes usar el buscador para encontrar más información.</small></div>
                </div>
                <aside class="hero-card">
                    <h3 style="margin-top:0">Acerca</h3>
                    <p class="muted">Interfaz inspirada en la paleta y el diseño de la aplicación de Rick y Morty</p>
                </aside>
            </div>
        </section>
        <div class="container">
            <div id="grid" class="grid" aria-live="polite"></div>
        </div>
    </main>
    <footer class="site-footer">
        <div class="container">
            <div class="muted">Datos proporcionados por <a href="https://rickandmortyapi.com/">rickandmortyapi.com</a></div>
        </div>
    </footer>
    <script src="code.js"></script>
</body>
```

Semana 11 – Reporte

Funciones principales:

meta charset="UTF-8": Permite el uso de caracteres especiales.

meta viewport: Hace que el sitio sea adaptable a dispositivos móviles.

link rel="stylesheet": Importa el archivo style.css, encargado del diseño visual.

title: Define el título que aparece en la pestaña del navegador.

Elementos:

Título "PRODUCT LIST": Indica la sección donde se mostrarán los productos.

Icono del carrito: Representado mediante un SVG, tomado de Flowbite Icons.

Contador (0): Muestra la cantidad de productos agregados al carrito.

A screenshot of a Microsoft Edge browser window. The left side shows a code editor with an HTML file named 'index.html'. The code includes a header section with meta tags and a title, a 'container' div containing a 'header' with a logo and a 'listProduct' section, and a 'cartTab' section with a 'Shopping Cart' heading and a 'listCart' section. The right side features the 'Welcome to Copilot' interface from GitHub, which includes a sidebar with options like 'Add context (F)', 'extensions (I)', 'commands', 'Build Workspace', and 'Show Config'. Below the sidebar is a message to 'Review AI output carefully before use.'

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- Importar el estilo.css -->
    <link rel="stylesheet" href="style.css">
    <title>Proyecto E-commerce</title>
</head>
<body>
    <div class="container">
        <header>
            <div class="title">PRODUCT LIST</div>
            <div class="icon-cart">
                <!-- Referencia del icono https://flowbite.com/icons/ -->
                <img alt="Icono de carrito de compras" class="w-6 h-6 text-gray-900 dark:text-white" aria-label="Cart icon"/>
                <span>4</span>
            </div>
        </header>

        <div class="listProduct">
            <div class="item">
                ...
            </div>
        </div>
    </div>

    <div class="cartTab">
        <h2>Shopping Cart</h2>
        <div class="listCart">
            ...
        </div>
    </div>
</body>
</html>
```

A second screenshot of a Microsoft Edge browser window, similar to the first one. It shows the same 'index.html' file with a different code structure. The 'listCart' section now contains its own 'item' divs, and there is a new 'btn' div with 'CLOSE' and 'CHECK OUT' buttons. The rest of the code remains the same as the first screenshot.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- Importar el estilo.css -->
    <link rel="stylesheet" href="style.css">
    <title>Proyecto E-commerce</title>
</head>
<body>
    <div class="container">
        <header>
            ...
        </header>

        <div class="listProduct">
            <div class="item">
                ...
            </div>
        </div>
    </div>

    <div class="cartTab">
        <h2>Shopping Cart</h2>
        <div class="listCart">
            <div class="item">
                ...
            </div>
        </div>
        <div class="btn">
            <button class="close">CLOSE</button>
            <button class="checkout">CHECK OUT</button>
        </div>
    </div>
</body>
</html>
```

Semana 12 – Reporte

El presente código CSS tiene como finalidad definir el diseño visual, la estructura y el comportamiento responsivo de una aplicación web. El estilo está orientado a mostrar elementos tipo tarjetas (cards), botones y listas, haciendo uso de variables CSS, flexbox, grid layout y media queries para lograr una interfaz moderna, ordenada y adaptable a distintos tamaños de pantalla.

Se realiza un reset de estilos, eliminando márgenes y rellenos por defecto, y estableciendo:

box-sizing: border-box para un mejor control del tamaño de los elementos

Fuente Rubik como tipografía principal

Estilos generales de elementos

ul: elimina viñetas de listas

button: elimina bordes y fondos por defecto

body: establece altura mínima completa y color de fondo gris

Información interna del Pokémon

Se definen estilos para:

Imagen (.pokemon-imagen)

Identificador (.pokemon-id)

Nombre (.pokemon-nombre)

Tipos (.pokemon-tipos)

Estadísticas (.pokemon-stats)

Todo se organiza usando Flexbox para centrar y distribuir el contenido correctamente.

The screenshot shows a code editor with two tabs: 'index.html' and 'code.js'. The 'index.html' tab contains the following CSS code:

```
1  @import url('https://fonts.googleapis.com/css2?family=Rubik:wght@300;400');
2
3  :root {
4      --clr-black: #1c1c1c;
5      --clr-gray: #e0e0e0;
6      --clr-white: #f7f7f7;
7
8      --type-normal: #A8A878;
9      --type-fire: #F08030;
10     --type-water: #6B00F0;
11     --type-grass: #7BC850;
12     --type-electric: #F8D030;
13     --type-ice: #98D8D8;
14     --type-fighting: #C0392B;
15     --type-poison: #A8M8A0;
16     --type-ground: #E0C068;
17     --type-flying: #A8B8F0;
18     --type-psychic: #F0A8B8;
19     --type-bug: #A8B820;
20     --type-rock: #B8A8A0;
21     --type-ghost: #705898;
22     --type-dark: #705848;
23     --type-dragon: #7038FB;
24     --type-steel: #B8B8D0;
25     --type-fairy: #F0B8B8;
26 }
27
28 * {
29     margin: 0;
30     padding: 0;
31     box-sizing: border-box;
32     color: var(--clr-black);
33 }
```

The 'code.js' tab has the number '9' next to it. On the right side of the screen, there is a 'Welcome to Copilot' interface with a logo, a 'Let's get started' button, and a text input field for adding context, extensions, and commands. Below that are buttons for 'Build Workspace' and 'Show Config', and a note to review AI output carefully before use. The bottom of the screen shows a taskbar with various icons and system status information.

The screenshot shows a code editor interface with a dark theme. On the left, there is a file tree with a single item: 'index.html'. The main area contains the following CSS code:

```
index.html
1 .header {
2     display: flex;
3     align-items: center;
4     flex-wrap: wrap;
5     gap: .5rem;
6 }
7
8 .btn-header {
9     background-color: var(--clr-gray);
10    padding: .5rem;
11    border-radius: 100vmax;
12    cursor: pointer;
13    text-transform: uppercase;
14    font-weight: 600;
15    box-shadow: 0 0 1rem rgba(0, 0, 0, 1);
16    transition: .2s;
17 }
18
19 .btn-header:hover {
20     transform: scale(1.1);
21     box-shadow: 0 0 2rem rgba(0, 0, 0, 1);
22 }
23
24 main {
25     padding: 2rem;
26     max-width: 1000px;
27     margin: 0 auto;
28 }
29
30
31 /*Este código permite validar cuando se visualice en una pantalla de cel
32 .pokemon-todos{
```

On the right side of the screen, there is a 'Copilot' panel titled 'Welcome to Copilot'. It features a small AI icon, a 'Let's get started' button, and a text input field with placeholder text: 'Add context (F), extensions (B), commands (C)'. Below this is a 'Build Workspace' button and a 'Show Config' button. A note at the bottom says 'Review AI output carefully before use.' The status bar at the bottom of the editor shows 'Ln 295, Col 2' and other standard file and encoding information.

Semana 13 – Reporte

EduTrack - Sistema de Asistencia y Participación

Descripción general

EduTrack es un sistema completo de gestión de asistencia y participación estudiantil diseñado para instituciones educativas. Ofrece una interfaz moderna y eficiente que permite a los docentes registrar, monitorear y analizar el rendimiento académico de sus estudiantes.

Características principales

Gestión de Asistencia

Registro rápido de asistencia por clase y fecha

Marcación masiva (todos presentes/ausentes)

Registro de llegadas tardías

Historia completa de asistencia por estudiante.

Seguimiento de Participación

Evaluación de participación con sistema de puntuación (1-10)

Categorización por tipo de participación (respuesta, pregunta, participación activa, presentación)

Notas descriptivas para cada registro

Ánalisis de patrones de participación

Reportes y Análisis

Dashboard con estadísticas en tiempo real

Informes detallados de asistencia y participación

Ánalisis individual de estudiantes

Vista general por clase

Exportación de datos en formato JSON

Interfaz moderna

Diseño responsivo y accesible

Animaciones suaves y microinteracciones.

Panel de control intuitivo

Notificaciones en tiempo real

Arquitectura del Sistema

Tecnologías utilizadas

Interfaz : HTML5, CSS3, JavaScript ES6+

Marco CSS : Tailwind CSS

Animaciones : Anime.js

Almacenamiento : LocalStorage (base de datos local)

Tipografías : Inter (texto principal), JetBrains Mono (datos)

Welcome index.html code.js

index.html (HTML)

```
<!DOCTYPE html><html lang="es"><head>
    <meta charset="UTF-8"/>
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
    <title>Edutrack - Sistema de Asistencia y Participación</title>
    <script src="https://cdn.tailwindcss.com/ajax/libs/tailwindcss/3.2.1/tailwind.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/animatic/3.3.1/animate.min.js"></script>
    <link rel="preconnect" href="https://fonts.googleapis.com"/>
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin="anonymous"/>
    <link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;500;700;900;900italic" rel="stylesheet"/>
<style>
    * { font-family: 'Inter', sans-serif; }
    .mono { font-family: 'JetBrains Mono', monospace; }
    .gradient-bg { background: linear-gradient(135deg, #efafaf 0%, #f0f0f0 100%); }
    .card-hover { transition: all 0.2s ease-out; }
    .card-hover:hover { transform: translateY(-2px); box-shadow: 0 3px 10px #f0f0f0; }
    .pulse-animation { animation: pulse 2s infinite; }
    @keyframes pulse { 0%, 100% { opacity: 1; } 50% { opacity: 0.5; } }
    .slide-in { animation: slideIn 0.3s ease-out; }
    @keyframes slideIn { from { transform: translateY(-20px); opacity: 0; } to { transform: translateY(0); opacity: 1; } }
    .loading-skeleton { background: linear-gradient(90deg, #f0f0f0 0%, #e0e0e0 100%); }
    @keyframes loading { 0% { background-position: 200% 0; } 100% { background-position: 0 0; } }
</style>
<script src="https://statics.moonshot.cn/sdk/preview-widgets.min.js" defer></script>
<body class="gradient-bg min-h-screen">
    <!-- Navigation -->
    <nav class="bg-white shadow-sm border-b border-gray-200 sticky top-0 w-full px-4 py-2">
        <div class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8">
            <div class="flex justify-between items-center h-16">
                <div class="flex items-center space-x-4">
                    <div class="flex-shrink-0">
                        <img alt="Edutrack logo" class="text-2xl font-bold text-blue-600" />
                    </div>
```

Welcome index.html • JS code.js 9+ • CHAT + ⏱ ⏴ ⏵ ⏷ | [] x

index.html > html

```
1  <!DOCTYPE html><html lang="es"><head>
2      <body class="gradient-bg min-h-screen">
3          <div class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8 py-8">
4              <div class="grid grid-cols-1 lg:grid-cols-3 gap-6 mb-8">
5                  <div class="bg-white rounded-xl p-6 shadow-sm border border-gray-200">
6                      <div id="studentList" class="space-y-3 max-h-72 overflow-y-hidden">
7                          <!-- Student list will be populated by JavaScript -->
8                  </div>
9              </div>
10             </div>
11
12             <!-- Main Actions -->
13             <div class="grid grid-cols-1 lg:grid-cols-3 gap-8">
14                 <!-- Attendance Section -->
15                 <div class="bg-white rounded-xl shadow-sm border border-gray-200">
16                     <div class="p-6 border-b border-gray-200">
17                         <h3 class="text-lg font-semibold text-gray-900 mb-2">Marca la asistencia
18                         <p class="text-sm text-gray-600">Marca la asistencia
19                     </div>
20                     <div class="p-6">
21                         <div class="space-y-4">
22                             <div>
23                                 <label class="block text-sm font-medium text-gray-900">Clase:</label>
24                                 <select id="classSelect" class="w-full border border-gray-200">
25                                     <option value="">Seleccionar clase</option>
26                                     <option value="math-10a">Matemáticas 10A</option>
27                                     <option value="science-10b">Ciencias 10B</option>
28                                     <option value="history-9a">Historia 9A</option>
29                                 </select>
30                             </div>
31                         </div>
32                     </div>
33                 </div>
34             </div>
35         </div>
36     </body>
37 </html>
```

Ln 328, Col 15 Spaces: 4 UTF-8 CRLF (HTML Go Live Prettier)

Búsqueda

CHAT + ⏱ ⏴ ⏵ ⏷ | [] x

Welcome to Copilot

Let's get started

Add context (#), extensions (@), commands ➔

Build Workspace Show Config

Review AI output carefully before use.

Semana 14 – Reporte

Registro de los datos

Tener un arreglo, dentro de ese arreglo una estructura los datos que solicita son una id, título, descripción, completada.

El presente código HTML corresponde a una aplicación web tipo ToDo, cuyo objetivo principal es permitir al usuario registrar, consultar, editar y eliminar tareas, utilizando almacenamiento local del navegador. El archivo define la estructura base de la interfaz, mientras que el diseño y la lógica se implementan mediante archivos externos de CSS y JavaScript.

Función de los elementos:

meta charset="UTF-8": Permite el uso de caracteres especiales.

meta viewport: Garantiza un diseño responsivo.

title: Define el nombre de la página en el navegador.

link stylesheet: Conecta el archivo CSS para el diseño visual.

meta description: Describe brevemente la funcionalidad de la aplicación.

El archivo app.js se encarga de:

- Gestionar eventos del formulario
- Almacenar tareas en localStorage
- Mostrar, editar y eliminar tareas
- Controlar el modal de edición

NetBeansProjects > Alumnos > index.html > index.html

```
3 <html lang="es">
4   <body>
5     <main class="container">
6       <section class="panel">
7         <div class="controls">
8           </div>
9
10          <ul id="task-list" class="task-list" aria-live="polite"></ul>
11        </section>
12
13        <!-- Edit modal -->
14        <div id="edit-modal" class="modal hidden" role="dialog" aria-modal="true">
15          <div class="modal-content">
16            <h3 id="edit-title">Editar tarea</h3>
17            <form id="edit-form" class="form">
18              <input type="hidden" id="edit-id">
19              <label for="edit-title-input">Título:</label>
20              <input type="text" id="edit-title-input" required>
21
22              <label for="edit-description-input">Descripción:</label>
23              <textarea id="edit-description-input"></textarea>
24
25              <label class="checkbox"><input type="checkbox" id="e>
26
27              <div class="form-actions">
28                <button type="submit" class="btn">Guardar</button>
29                <button type="button" id="edit-cancel" class="bt>
30              </div>
31            </form>
32          </div>
33        </div>
34      </main>
```

In 73, Col 8 Spaces 4 UTF-8 CRLF (A) HTML (B) Go Live (C) Prettier

Búsqueda Buscador de archivos

ESP LAA 07:40 p.m. 13/12/2023

NetBeansProjects > Alumnos > index.html > index.html

```
3 <html lang="es">
4   <body>
5     <main class="container">
6       <section class="panel">
7         <div class="controls">
8           </div>
9
10          <ul id="task-list" class="task-list" aria-live="polite"></ul>
11        </section>
12
13        <!-- Edit modal -->
14        <div id="edit-modal" class="modal hidden" role="dialog" aria-modal="true">
15          <div class="modal-content">
16            <h3 id="edit-title">Editar tarea</h3>
17            <form id="edit-form" class="form">
18              <input type="hidden" id="edit-id">
19              <label for="edit-title-input">Título:</label>
20              <input type="text" id="edit-title-input" required>
21
22              <label for="edit-description-input">Descripción:</label>
23              <textarea id="edit-description-input"></textarea>
24
25              <label class="checkbox"><input type="checkbox" id="e>
26
27              <div class="form-actions">
28                <button type="submit" class="btn">Guardar</button>
29                <button type="button" id="edit-cancel" class="bt>
30              </div>
31            </form>
32          </div>
33        </div>
34      </main>
```

In 73, Col 8 Spaces 4 UTF-8 CRLF (A) HTML (B) Go Live (C) Prettier

Búsqueda Buscador de archivos

ESP LAA 07:40 p.m. 13/12/2023

Semana 15 – Reporte

El presente código tiene como finalidad implementar una estructura de datos tipo Tabla Hash (Hash Table) utilizando clases en JavaScript. Esta estructura permite almacenar y recuperar datos mediante pares clave–valor, ofreciendo un acceso eficiente a la información a través de una función hash.

El programa define una clase llamada HashTable, la cual:

Utiliza un arreglo como estructura base.

Emplea una función hash para convertir claves en índices.

Implementa métodos para insertar, obtener, eliminar y listar claves.

```
hashMethod(key) {  
    let hash = 0;  
    for (let i = 0; i < key.length; i++) {  
        hash = (hash + key.charCodeAt(i) * i) % this.data.length;  
    }  
    return hash;  
}
```

Este método:

Convierte una clave tipo string en un índice numérico.

Utiliza el valor ASCII de cada carácter.

Aplica el operador módulo para asegurar que el índice esté dentro del tamaño del arreglo.

```
set(key, value) {  
    const address = this.hashMethod(key);  
  
    if (!this.data[address]) {  
        this.data[address] = [];  
    }  
  
    this.data[address].push([key, value]);  
  
    return this.data;  
}
```

Permite insertar un par clave–valor en la tabla hash.

Si ya existe información en esa posición, se utiliza un arreglo interno (bucket) para manejar colisiones mediante separate chaining.

The screenshot shows a code editor with the following details:

- Header:** Welcome, Untitled-1, JS code.js, index.html, ...
- Breadcrumbs:** etBeansProjects > Alumnos > JS code.js > ...
- Code Content:** A class definition for HashTable. The code includes methods for setting and getting values based on a hash key.

```
1  class HashTable {
2      constructor(size) {
3          this.data = new Array(size);
4      }
5
6      hashMethod(key) {
7          let hash = 0;
8          for (let i = 0; i < key.length; i++) {
9              hash = (hash + key.charCodeAt(i) * i) % this.data.length;
10         }
11         return hash;
12     }
13
14     set(key, value) {
15         const address = this.hashMethod(key);
16         if (!this.data[address]) {
17             this.data[address] = [];
18         }
19         this.data[address].push([key, value])
20         return this.data;
21     }
22
23     get(key) {
24         const address = this.hashMethod(key);
25         const currentBucket = this.data[address];
26         if (currentBucket) {
27             for (let i = 0; i < currentBucket.length; i++) {
28                 if (currentBucket[i][0] === key) {
29                     return currentBucket[i][1];
30                 }
31             }
32         }
33     }
34 }
```

- Bottom Status Bar:** Ln 65, Col 39, Spaces: 4, UTF-8, CRLF

Semana 16 – Reporte

El presente código HTML corresponde a la estructura base de una aplicación web que tiene como finalidad consumir y mostrar información de la API de Pokémon, permitiendo filtrar los resultados por tipo. El archivo define la organización visual de la página y se apoya en un archivo CSS externo para el diseño.

Función de los elementos:

meta charset="UTF-8": Permite la correcta visualización de caracteres especiales.

meta viewport: Asegura un diseño responsivo para dispositivos móviles.

title: Define el título de la página en el navegador.

link rel="stylesheet": Conecta el archivo index.css, encargado del diseño visual.

```
<main>
    <h1>Hola como estas</h1>
    <p>Hola como estas</p>
</main>
```

Esta sección representa el contenido principal de la página. Actualmente contiene texto de ejemplo, el cual puede ser reemplazado por:

Tarjetas de Pokémon

Información obtenida dinámicamente desde la API

Resultados del filtrado por tipo

Integración con CSS y JavaScript

Aunque el código presentado solo incluye HTML y CSS:

El archivo index.css se encarga del diseño visual.

Posteriormente puede integrarse un archivo JavaScript para:

Consumir la API de Pokémon

Mostrar los datos en pantalla

Filtrar resultados según el botón seleccionado

The screenshot shows a Microsoft Edge browser window. On the left, the code editor displays an HTML file with the following content:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<body>
    <div class="container-type">
        <button class="btn-type-normal">ver todos</button>
        <button class="btn-type-normal">Normal</button>
        <button class="btn-type-fire">Fire</button>
        <button class="btn-type-water">Water</button>
        <button class="btn-type-grass">Grass</button>
        <button class="btn-type-electric">Electric</button>
        <button class="btn-type-ice">Ice</button>
        <button class="btn-type-fighting">Fighting</button>
        <button class="btn-type-poison">Poison</button>
        <button class="btn-type-ground">Ground</button>
        <button class="btn-type-flying">Flying</button>
        <button class="btn-type-psychic">Psychic</button>
        <button class="btn-type-bug">Bug</button>
        <button class="btn-type-rock">Rock</button>
        <button class="btn-type-ghost">Ghost</button>
        <button class="btn-type-dragon">Dragon</button>
        <button class="btn-type-dark">Dark</button>
        <button class="btn-type-steel">Steel</button>
        <button class="btn-type-fairy">Fairy</button>
    </div>
</head>
<body>
    <h1>Hola como estas!</h1>
    <p>Hola como estas!</p>
</body>
</html>
```

On the right, the Copilot AI panel is open with the title "Welcome to Copilot". It includes a "Let's get started" button, a "Add context (P), extensions (B), commands" input field, and buttons for "Build Workspace" and "Show Config". A note at the bottom says "Review AI output carefully before use".

INSIGNIA

https://developers.google.com/profile/u/104498823974458677367?utm_source=web.dev

